



UNIVERSIDADE DO VALE DO TAQUARI - UNIVATES
PROGRAMA DE PÓS-GRADUAÇÃO *STRICTO SENSU*
MESTRADO EM ENSINO

**O *SOFTWARE VISUALG* COMO RECURSO DIDÁTICO
NO ENSINO DA LÓGICA DE PROGRAMAÇÃO**

Fabício Hartmann Borba

Lajeado, junho de 2018

Fabício Hartmann Borba

**O *SOFTWARE VISUALG* COMO RECURSO DIDÁTICO
NO ENSINO DA LÓGICA DE PROGRAMAÇÃO**

Dissertação apresentada ao Programa de Pós-Graduação em Ensino, da Universidade do Vale do Taquari - Univates, como parte da exigência para a obtenção do grau de Mestre em Ensino, na linha de pesquisa Recursos, Tecnologias e Ferramentas no Ensino.

Orientadora: Profa. Dra. Miriam Ines Marchi

Coorientadora: Profa. Dra. Márcia Jussara Hepp Rehfeldt

Lajeado, junho de 2018

Fabício Hartmann Borba

O *SOFTWARE VISUALG* COMO RECURSO DIDÁTICO NO ENSINO DA LÓGICA DE PROGRAMAÇÃO

A Banca examinadora aprova a Dissertação apresentada ao Programa de Pós-Graduação em Ensino, da Universidade do Vale do Taquari - Univates, como parte da exigência para a obtenção do grau de Mestre em Ensino, na linha de pesquisa Recursos, Tecnologias e Ferramentas no Ensino.

Profa. Dra. Miriam Ines Marchi - Orientadora
Universidade do Vale do Taquari - Univates

Profa. Dra. Márcia Jussara Hepp Rehfeldt - Coorientadora
Universidade do Vale do Taquari - Univates

Profa. Dra. Ieda Maria Giongo
Universidade do Vale do Taquari - Univates

Prof. Dr. Italo Gabriel Neide
Universidade do Vale do Taquari - Univates

Prof. Dr. Rogério José Schuck
Universidade do Vale do Taquari - Univates

Lajeado, junho de 2018

RESUMO

A crescente demanda de profissionais de Tecnologia da Informação (TI) em empresas, universidades e outras instituições é notória e, para tal, é necessária a formação de programadores para darem suporte a criação e manutenção de sistemas computacionais. O algoritmo, que é a base para criação dos diversos software e aplicativos, pode ser ensinado de forma a facilitar o aprendizado dos estudantes, ao utilizar o Português Estruturado. Em função disso, explorou-se uma prática pedagógica com a participação dos sete estudantes, da disciplina de Algoritmos, do Módulo III - Assistente em Programação, do curso Técnico em Informática, da Escola Estadual de Educação Profissional Estrela. Foram ministradas cinco aulas, nas quais utilizou-se o *Software VisuAlg* como ferramenta auxiliar no ensino da estrutura dos algoritmos, com a criação de códigos em Português. O principal objetivo foi analisar as potencialidades do *Software VisuAlg* como recurso didático nas aulas iniciais da disciplina de Algoritmos. Os dados foram coletados mediante dois questionários e por meio de anotações no diário de campo, no qual registrou-se as aulas desenvolvidas, as atividades, as falas de alguns estudantes, além do registro por gravação de áudio e dos algoritmos criados. Os dados coletados foram analisados com aproximações à Análise Textual Discursiva (MORAES; GALIAZZI, 2013). Classificou-se os dados em categorias contemplando os objetivos específicos: 1) Conhecimentos sobre algoritmos - advinda da análise das respostas ao “Questionário de Sondagem”; 2) Contribuições do *Software VisuAlg* no ensino de algoritmos - resultante da análise dos dados registrados no diário de campo; 3) Percepções dos estudantes sobre o *Software VisuAlg* - proveniente da análise das respostas apresentadas no “Questionário - Avaliação da Prática Pedagógica”. Os resultados encontrados apontam que: a) a utilização de diferentes técnicas e ferramentas auxilia no ensino da lógica de programação, pois instiga o estudante a experimentar e testar os algoritmos; b) a criação de algoritmos na Língua Portuguesa ajuda os estudantes a pensarem em sua lógica, não sendo necessário compreender comandos em inglês; c) a motivação dos estudantes aumenta, pois pode-se comprovar que os códigos criados estão funcionando, além da facilidade de poder encontrar os erros. Portanto, a utilização de ferramentas que auxiliem no ensino da lógica de programação, e estimulem o interesse dos estudantes nesta área, é cada vez mais utilizada por diferentes professores.

Palavras-chave: Ensino. Lógica de Programação. Algoritmos. Português Estruturado. *Software VisuAlg*.

ABSTRACT

The growing demand for professionals of Information Technology (IT) in companies, universities and other institutions is notorious, and for that, it is necessary the formation of programmers to creation support and maintenance of computer systems. The algorithm, which is the basis for the creation of several software and applications, can be taught in a way that facilitates student's learning, using Structured Portuguese. Because of this, a pedagogical practice was explored with the participation of the seven students, from the Algorithms discipline, from Module III - Assistant in Programming, from the Technical Course in Computing, from the Escola Estadual de Educação Profissional Estrela. Five classes were taught, in which the Software VisuAlg was used as an auxiliary tool in teaching the structure of the algorithms, with the creation of codes in Portuguese. The main objective was to analyze the potential of the Software VisuAlg as didactic resource in the initial classes of the Algorithms discipline. The data were collected by two questionnaires and by means of annotations in the field diary, in which the developed classes, the activities, the speeches of some students were registered, as well as the audio recordings and the codes of the created algorithms. The collected data were analyzed with approximations to the Discursive Textual Analysis (MORAES; GALIAZZI, 2013). The data were categorized according to the specific objectives: 1) Previous knowledge - from the analysis of the answers to the "Survey Questionnaire"; 2) VisuAlg Software contributions in teaching algorithms - resulting from the analysis of the data recorded in the field diary; 3) Students perceptions about VisuAlg Software - from the analysis of the answers presented in the "Questionnaire - Evaluation of Pedagogical Practice". The results show that: a) the use of different techniques and tools helped in the teaching of programming logic, as it instigates students to experiment and test the algorithms; b) the creation of algorithms in the Portuguese Language helped students to think in their logic, not being necessary to understand commands in English; c) the motivation of the students increase, because it can be verify that the codes created are working, as well as the facility of being able to find the errors. Therefore, the use of tools that help in teaching the programming logic, and stimulate the interest of students in this area, is increasingly used by different teachers.

Keywords: Teaching. Programming Logic. Algorithms. Structured Portuguese. Software VisuAlg.

LISTA DE FIGURAS

Figura 01 - Esquema de ensino e de aprendizagem por meio do computador	18
Figura 02 - Diagrama do processo de programação	25
Figura 03 - Exemplo algoritmo escrito em português estruturado	27
Figura 04 - Tela inicial do <i>Software VisuAlg</i>	30
Figura 05 - Algoritmo escrito no <i>Software VisuAlg</i>	31
Figura 06 - Tela de execução do código da Figura 05	32
Figura 07 - Comparativo entre Português Estruturado e a Linguagem <i>Java</i>	32
Figura 08 - Exercícios sobre descrição narrativa	59
Figura 09 - Representação gráfica do Exercício A feita em fluxograma	60
Figura 10 - Fluxograma mais completo do Exercício A	61
Figura 11 - Comparação entre Descrição Narrativa, Fluxograma e Português Estruturado	62
Figura 12 - Exercícios sobre fluxograma	63
Figura 13 - Resposta do Estudante E para a Questão G	65
Figura 14 - Algoritmo troca de valores	66
Figura 15 - Exercícios sobre SE utilizando o <i>Software VisuAlg</i>	68
Figura 16 - Algoritmo do Estudante D com erro no teste de três opções de resposta	69
Figura 17 - Algoritmo do Estudante D após a correção	69
Figura 18 - Mensagem de erro: Esperava encontrar FIMSE	71
Figura 19 - Algoritmo “Testando”, criado pelos estudantes	72
Figura 20 - Laço de Repetição PARA executando passo a passo	73
Figura 21 - Comparação entre o PARA e o ENQUANTO	74
Figura 22 - Exercício de Laço de Repetição PARA com erro	74

Figura 23 - Mensagem de erro do Laço de Repetição PARA	75
Figura 24 - Exercício de Laço de Repetição ENQUANTO com erro	76
Figura 25 - Mensagem de erro do Laço de Repetição ENQUANTO	76
Figura 26 - Questão 08, Questionário - Avaliação da Prática Pedagógica.....	82

LISTA DE QUADROS

Quadro 01 - Exemplo de algoritmo comparado com linguagem de programação.....	23
Quadro 02 - Atividades realizadas durante a prática pedagógica	47

LISTA DE ABREVIATURAS E SIGLAS

AMBAP	Ambiente de Aprendizado de Programação
ASA	Animação e Simulação de Algoritmos
CAPES	Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
EEEPE	Escola Estadual de Educação Profissional Estrela
FECITEP	Feira Estadual de Ciência e Tecnologia da Educação Profissional
HTML	<i>HyperText Markup Language</i>
IDE	<i>Integrated Development Environment</i>
ISO	<i>International Organization for Standardization</i>
MEC	Ministério da Educação
MEEP	Mostra de Ensino, Extensão e Pesquisa da UNIVATES
MEP	Mostra das Escolas de Educação Profissional, Ensino Médio Politécnico e Curso Normal
MOSTRATEC	Mostra Brasileira de Ciência e Tecnologia e Mostra Internacional de Ciência e Tecnologia
PHP	PHP: <i>Hypertext Preprocessor</i>
RJ	Rio de Janeiro
TI	Tecnologia da Informação
UNIVATES	Universidade do Vale do Taquari
<i>VisuAlg</i>	Visualizador de Algoritmos

SUMÁRIO

1 INTRODUÇÃO	10
2 REFERENCIAL TEÓRICO	15
2.1 Sociedade e Tecnologia	15
2.2 Algoritmos e Programação	20
2.3 Português Estruturado e <i>VisuAlg</i>	26
2.4 Trabalhos acerca do ensino de algoritmos.....	33
3 PROCEDIMENTOS METODOLÓGICOS.....	41
3.1 Caracterização da pesquisa	41
3.2 Organização da Pesquisa	43
3.3 Instrumentos utilizados para coleta de dados	44
3.4 Estrutura da prática pedagógica.....	47
3.5 Curso Técnico em Informática da Escola Estadual de Educação Profissional Estrela	49
3.5.1 Caracterização da turma	50
3.6 Categorização dos dados.....	50
4 ANÁLISE E RESULTADOS.....	53
4.1 Categoria 1 - Conhecimentos sobre algoritmos	53
4.2 Categoria 2 - Contribuições do Software <i>VisuAlg</i> no ensino de algoritmos.....	57
4.3 Categoria 3 - Percepções dos estudantes sobre o Software <i>VisuAlg</i>	78
5 CONSIDERAÇÕES FINAIS	85
REFERÊNCIAS.....	90
APÊNDICES.....	95

1 INTRODUÇÃO

Cada vez mais são necessários profissionais da área de informática e computação em empresas, universidades e outras instituições de diversos setores da sociedade. Isso deve-se a crescente informatização e automatização de serviços que, antigamente, eram realizados de forma manual. Portanto, pessoas com habilidades e conhecimentos são indispensáveis para desenvolver e dar suporte a máquinas e sistemas que supram o crescente avanço tecnológico e a inevitável modernização das organizações.

Diante deste contexto, o presente estudo, vinculado ao Programa de Pós-graduação Mestrado em Ensino da Universidade do Vale do Taquari - Univates, inserido na linha de pesquisa “Recursos, Tecnologias e Ferramentas no Ensino”, está fundamentado em pesquisas acerca da inserção e da utilização de ferramentas como recurso didático para auxiliar na construção da lógica de programação. Esta lógica é necessária para a construção da estrutura dos algoritmos, utilizados em disciplinas de Algoritmos e Programação de Computadores, que são a base para o desenvolvimento de sistemas computadorizados.

Neste sentido, foram analisadas as contribuições do uso do *Software VisuAlg*, que utiliza a Língua Portuguesa para a escrita dos algoritmos. Esta forma de escrita é conhecida como Português Estruturado, Portugol ou Pseudocódigo, pois todos os comandos e funções podem ser descritos em português e são adaptações genéricas da linguagem de programação. Dessa forma, ao utilizar a língua materna dos estudantes entende-se que isso pode contribuir para a compreensão da lógica inicial

necessária para a resolução de problemas computacionais, conforme destaca Carvalho (2007).

O principal estímulo para pesquisar sobre este tema iniciou-se com a atualização pedagógica do autor deste estudo, realizada por meio da graduação em Licenciatura em Computação, e pelas experiências empíricas como Professor de Informática nas disciplinas de Algoritmos e Programação de Computadores, com turmas do curso Técnico em Informática, entre os anos de 2010 e 2015. Ao analisar as aulas ministradas e o desempenho dos estudantes ao resolverem os algoritmos propostos, percebeu-se que alguns estudantes precisavam de um tempo maior, além de exercitar a resolução de problemas computacionais¹, para compreender a lógica de programação, bem como o processo de escrita dos algoritmos apresentados em aula.

Além disso, outra situação que se observou, durante as aulas ministradas na disciplina de Algoritmos, é que os códigos eram desenvolvidos com “papel e caneta” e os estudantes apresentavam dificuldades em compreender qual o resultado do algoritmo construído. Supunha-se que isso pudesse estar relacionado ao fato de ser um método de construção abstrato, no qual o estudante precisa entender a lógica do que é escrito para “prever” o resultado final. Em outras palavras, por meio da utilização do “teste de mesa”² o estudante tentava comprovar se a construção apresentava-se correta ou não.

Acredita-se que, além dos motivos citados, outro que também pode ser considerado como obstáculo para a compreensão da lógica de programação, é que as linguagens de programação são geralmente escritas na Língua Inglesa. Sabe-se que alguns estudantes podem não entender o significado de certos termos nesta língua.

Estas percepções também são apresentadas por Souza (2009), professor da Universidade Severino Sombra - RJ e do Centro de Ensino Superior de Valença - RJ, ao expor os objetivos que o levaram a criação do *Software VisuAlg*, em seu

¹ A resolução de problemas em computação consiste em analisar o enunciado das atividades propostas e criar um algoritmo que seja capaz de representá-lo (GUEDES, 2014).

² Teste de mesa - “Simula a execução de um algoritmo sem utilizar o computador, empregando apenas ‘papel e caneta’” (PIMENTEL, 2015, p. 3).

artigo intitulado “*VisuAlg* - Ferramenta de Apoio ao Ensino de Programação”. Souza (2009) descreve que, por meio de sua experiência, notou que escrever algoritmos no papel é abstrato demais para a compreensão de alguns estudantes. E que o contato inicial com alguma linguagem de programação na Língua Inglesa, que precisa de um rigor na criação do código, apresenta-se como uma barreira no aprendizado destes conceitos. Souza (2009, p. 2) cita ainda que se soma a isso “[...] o fato de que o aluno ainda enfrenta o obstáculo de entender o problema, criar por si uma solução ou compreender aquela apresentada por seus colegas ou pelo professor, isto é, a lógica de programação em si”.

Em outro estudo, Junior, Vieira e Vieira (2015) apresentam uma pesquisa realizada no Centro Universitário de Volta Redonda - RJ, em que alguns dos motivos que podem levar os estudantes a não compreenderem a lógica de programação, ou até reprovarem em disciplinas de Algoritmos e Programação são a dificuldade de raciocínio lógico necessário para a construção dos códigos, a capacidade de abstração e a leitura e interpretação dos textos dos enunciados. Em síntese, saber o que é preciso fazer e os conhecimentos matemáticos necessários para os testes e comparações que são realizados em algoritmos, são alguns dos motivos citados pelos autores.

Como se pode perceber, os motivos que levam os estudantes a apresentarem dificuldades na lógica de programação são percebidos não apenas em cursos técnicos, mas também, em cursos de nível superior nos quais os estudantes apresentam características semelhantes em relação à aprendizagem da programação. Neste sentido, Souza (2009) apresenta que diferentes professores e universidades já propõem meios, formas, técnicas e novos *softwares* que podem auxiliar no ensino da lógica de programação, entre os quais se destacam Ambiente de Aprendizado de Programação - AMBAP (Universidade Federal de Alagoas), Animação e Simulação de Algoritmos - ASA (Serviço Nacional de Aprendizagem Comercial), G-Portugol (União Pioneira de Integração Social), Portugol Studio (Universidade do Vale do Itajaí), Visualizador de Algoritmos - *VisuAlg* (Apoio Informática Ltda.).

Dessa forma, aprimorar as técnicas para o ensino da programação e criar *softwares* que auxiliem na construção da lógica dos estudantes podem ser fatores

importantes, pois com a crescente evolução e modernização da sociedade, são cada vez mais necessários programadores e desenvolvedores de sistemas em empresas e nas diferentes áreas acadêmicas e profissionais. Com isso, a utilização de instrumentos e ferramentas que auxiliem no ensino, na aprendizagem e estimulem cada vez mais o interesse dos estudantes nesta área são bem vindos aos professores.

Diante deste cenário, realizou-se a aplicação de uma prática pedagógica, na qual foi utilizado o *Software VisuAlg* para o ensino da estrutura dos algoritmos e a criação de códigos na Língua Portuguesa. Com isso, analisou-se a contribuição para o entendimento da lógica de programação nas aulas iniciais do curso Técnico em Informática, na disciplina de Algoritmos, do Módulo III - Assistente em Programação, da Escola Estadual de Educação Profissional Estrela.

Desta forma, apresenta-se a seguinte questão de pesquisa: **Como o *Software VisuAlg* pode ser utilizado de forma que seja um potente recurso didático para o ensino da lógica de programação na disciplina de Algoritmos?**

Diante da questão de pesquisa foi traçado o seguinte objetivo principal para ser desenvolvido na prática pedagógica: **Analisar as potencialidades do *Software VisuAlg* como recurso didático nas aulas da disciplina de Algoritmos, do curso Técnico em Informática, da Escola Estadual de Educação Profissional Estrela.**

Tem-se como objetivos específicos:

- Identificar os conhecimentos dos estudantes em relação à programação.
- Desenvolver uma proposta pedagógica, com estudantes do Módulo III do curso Técnico em Informática, que envolva a utilização do *Software VisuAlg* para a construção de algoritmos.
- Analisar como a utilização do *Software VisuAlg* contribui para a aprendizagem da lógica de programação.

Tendo em vista a questão inicial e os objetivos propostos, a pesquisa é considerada qualitativa e mantém aproximações com um estudo de caso. E foi por meio da prática pedagógica desenvolvida, com um grupo de estudantes na disciplina de Algoritmos, que investigou-se as contribuições que o *Software VisuAlg*

apresentou para o ensino da lógica de programação e para a construção de algoritmos na Língua Portuguesa.

Desse modo, a dissertação está estruturada em cinco capítulos, sendo o primeiro a introdução. Nela é descrita a trajetória profissional do autor deste estudo, os motivos que o levaram a pesquisa sobre este tema, a questão de pesquisa, o objetivo geral e os específicos.

O segundo capítulo apresenta a relação entre sociedade e tecnologia, o computador e os diferentes *softwares* que são utilizados no dia a dia das pessoas, a abordagem teórica sobre a lógica, a construção da lógica de programação durante o aprendizado dos estudantes, a fundamentação sobre algoritmos, português estruturado, o *Software VisuAlg* e, também, a apresentação de trabalhos recentes acerca do ensino de algoritmos.

Nesse seguimento, no terceiro capítulo é descrita a organização metodológica, os instrumentos utilizados na coleta dos dados, a caracterização da pesquisa, a estrutura e os materiais elaborados para a prática pedagógica, os procedimentos metodológicos utilizados, a caracterização da turma participante de prática pedagógica e a categorização dos dados.

Já no quarto capítulo é realizada a análise dos dados coletados na prática pedagógica, por meio dos questionários e anotações no diário de campo. A análise está dividida em três categorias, de acordo com os objetivos específicos: 1) Conhecimentos sobre algoritmos; 2) Contribuições do *Software VisuAlg* no ensino de algoritmos; e 3) Percepções dos estudantes sobre o *Software VisuAlg*.

Finalmente, no quinto capítulo, são expostas as considerações finais, que foram elaboradas a partir da análise dos resultados apontados nas categorias, considerando-se o referencial teórico e as percepções e experiências empíricas do autor sobre esta temática.

É neste contexto que vislumbra-se a possível contribuição deste estudo, que se propõe a analisar as potencialidades do uso do *Software VisuAlg* como recurso didático, nas aulas da disciplina de Algoritmos e Programação de Computadores. Portanto, no próximo capítulo é apresentado o referencial teórico desta pesquisa.

2 REFERENCIAL TEÓRICO

Neste capítulo é apresentado o referencial teórico que fundamenta esta pesquisa que contemplará os seguintes temas: sociedade e tecnologia, algoritmos e programação, português estruturado e *Software VisuAlg* e trabalhos acerca do ensino de algoritmos. A primeira seção trata sobre os avanços tecnológicos, o uso dos diferentes tipos de computadores e dos *softwares* que estão presentes no dia a dia das pessoas. Na segunda, é descrito sobre a lógica de programação necessária para a criação dos algoritmos essenciais para a programação. A terceira apresenta a fundamentação sobre o português estruturado e a escrita de algoritmos na Língua Portuguesa, utilizando o *Software VisuAlg*. E por fim, na quarta seção, são descritas algumas pesquisas que apresentam as dificuldades dos estudantes no ensino de algoritmos e sugestões de ferramentas e meios para tentar auxiliar na aprendizagem da lógica de programação, que foram utilizadas como ponto de partida no estudo dos algoritmos com o uso de *softwares* para auxiliar a aprendizagem.

2.1 Sociedade e Tecnologia

Pode-se perceber que com os avanços tecnológicos o acesso à informação está cada vez mais presente na vida das pessoas. Notícias que poderiam demorar dias ou até meses para chegar a diferentes lugares do mundo, hoje são visualizadas, por meio da *internet*, no momento em que acontecem. Küffer (2011, texto digital) apresenta que:

[...] os avanços tecnológicos em todas as áreas da sociedade, tendem a aumentar, uma vez que a existência de uma rede de informação e conhecimento composta por uma imensa capacidade de transformação e multiplicação, interfere diretamente no nosso cotidiano, chegando às redes virtuais e levando à expansão da informação e do conhecimento, eliminando barreiras geográficas e diminuindo as distâncias.

Com estes avanços tecnológicos, muitas são as mudanças que ocorreram e ainda acontecerão, em vários setores da sociedade. E na educação não é diferente, pois como comenta Schneider (2013, p. 99), “o homem do século XXI necessita desenvolver saberes e competências que lhe ajude a resolver problemas, a se relacionar e a inovar”, além de participar ativamente da sociedade.

Nesse sentido, cabe aqui mencionar que, o uso das tecnologias pode contribuir no desenvolvimento intelectual dos envolvidos, na qual professores e estudantes podem se beneficiar, por exemplo, dos diferentes tipos de *softwares* que estão disponíveis gratuitamente na *internet*, como auxiliares na construção de conhecimentos necessários para participarem como cidadãos ativos na sociedade. Além disso, com esses conhecimentos pode-se criar novas ferramentas e sistemas que possam contribuir ainda mais no desenvolvimento e no avanço tecnológico.

Dentre as diferentes tecnologias existentes, o computador apresenta várias possibilidades de utilização em sala de aula, como, por exemplo, o uso de programas matemáticos, mapas geográficos, vídeos, músicas, simuladores, entre outros de uso mais específico em diferentes assuntos trabalhados nas instituições de ensino. Com isso, os *softwares* podem apresentar-se como uma ferramenta na construção de conhecimentos básicos para uma sociedade em constante transformação e desenvolvimento.

A invenção do computador despertou diferentes sentimentos nas pessoas, dentre estes podem-se destacar, de acordo com Guedes (2014, p. VIII), que ele se tornaria “[...] o grande monstro sagrado, que dominaria o ser humano”. Após vários anos de sua criação, pode-se perceber que se tornou uma ferramenta, acessível a praticamente todas as pessoas e indispensável em diversos setores da sociedade. Além disso, Guedes (2014, p. VIII) destaca que:

[...] o computador finalmente se firmou como um objeto operacional para atender as necessidades humanas. Conecta-se agora aos seus irmãos cibernéticos, desde um simples celular até as gigantescas máquinas de previsão ambiental, de modo a constituir uma rede de conhecimentos, diversão e comunicação.

O computador pode ser considerado um acessório quase que indispensável na maioria das empresas e instituições de ensino, que mantêm seus dados e informações armazenadas em meio digital. Está presente também, no dia a dia das pessoas, em casa, no trabalho, na rua, nos bancos e em outros locais nos quais nem são percebidos. Dessa forma, convém ressaltar que, além do formato *desktop*³, o computador assumiu diferentes aparências, no qual os celulares, *smartphones*, *notebooks* e caixas eletrônicos, todos são considerados computadores, cada qual com suas especificações, funções e características, dependendo do propósito para o qual foi fabricado (GUEDES, 2014).

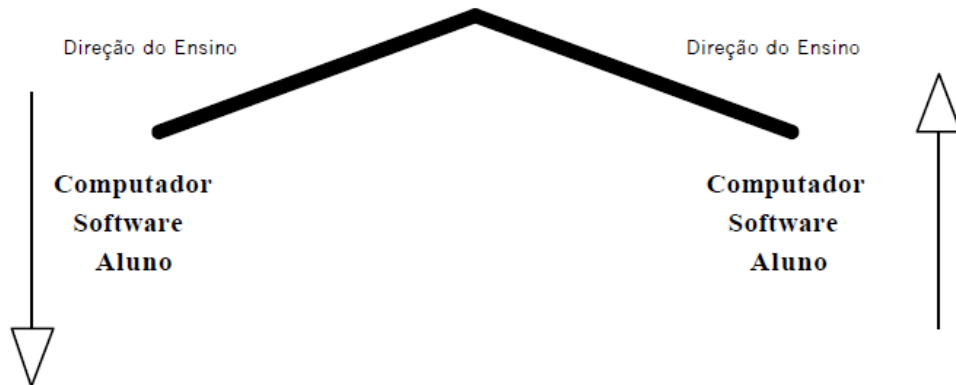
Contudo, para que os diferentes tipos de computadores possam operar, são necessários programas (*softwares*) para gerenciar o seu funcionamento, proporcionar entretenimento a quem os utiliza, trabalhar e auxiliar no processamento de cálculos que, segundo Guedes (2014), levariam uma eternidade para serem realizados pelas pessoas.

É importante saber que, para os computadores serem utilizados é preciso que sejam desenvolvidos os seus sistemas. Com isso, são necessárias pessoas com conhecimento técnico sobre a área de programação para desenvolver os programas. Celestino (2015) apresenta que os profissionais que são responsáveis por criarem os diferentes tipos de sistemas são chamados de programadores ou desenvolvedores. São eles que criam os sistemas, tornando, assim, os computadores acessíveis aos diferentes tipos de usuários. O que diferencia os desenvolvedores dos usuários é, principalmente, as suas funções em relação aos sistemas de computador. Os desenvolvedores criam sistemas para suprir as necessidades dos usuários em relação às várias possibilidades de uso do computador.

³ Desktop - “Literalmente, o termo tem o significado de ‘em cima da mesa’. Era frequentemente utilizado para designar um computador de mesa por oposição ao laptop que é o computador portátil” (SIGNIFICADOS, 2017, texto digital).

Assim, neste contexto, pode-se citar Valente (1999), que já nos anos 1990, apresentava a sua visão sobre as principais abordagens de como os computadores e *softwares* poderiam ser utilizados. Este autor fazia referência a sua utilização em sala de aula, mas pode-se abstrair deste contexto o que seriam as funções do usuário e do desenvolvedor. Valente (1993) trabalhava com basicamente duas abordagens, sendo que na primeira o estudante “aprende através do computador” e na segunda o computador é utilizado para “auxiliar na aprendizagem”. Uma representação destas abordagens é apresentada na Figura 01, que mostra o que Valente (1993) descreve sobre o ensino e a aprendizagem com o uso de computadores.

Figura 01 - Esquema de ensino e de aprendizagem por meio do computador.



Fonte: Valente (1993, p. 2).

Pode-se perceber que nas duas abordagens estão envolvidos os mesmos três personagens: computador, *software* e estudante, o que diferencia as duas é a maneira como o computador é utilizado. Valente (1993, p. 2) descreve essas abordagens de utilização do computador como tendo dois polos, no qual de um lado “[...] o computador, através do *software*, ensina o aluno. Enquanto no outro, o aluno, através do *software*, ‘ensina’ o computador”. Estas abordagens podem ser relacionadas com o usuário e o desenvolvedor, respectivamente.

Ao partir da abordagem de que o estudante tem a possibilidade, segundo Valente (1993, p. 3), de “ensinar” o computador a fazer ou resolver o que se propõe, pode, também, utilizar programas de linguagem computacional que “não prenda o usuário as suas funcionalidades, mas sim proporcione um ambiente para desenvolver as suas atividades com o auxílio do mesmo”. Com isso, o computador, por meio dos *softwares*, pode auxiliar no ensino e na aprendizagem e possibilitar o

desenvolvimento de conhecimentos para a construção da lógica de programação nos estudantes.

É com esta abordagem que estudantes de cursos de Informática e Computação podem criar sistemas e programas para auxiliar outras pessoas e empresas no uso do computador. Utilizam-se os *softwares* e, conseqüentemente o computador, para auxiliar na construção da lógica de programação, necessária para a criação de algoritmos capazes de solucionar problemas e, assim, posteriormente, o desenvolvimento de novos programas.

Para entender melhor o que são os programas, o Art. 1º, da Lei n. 9.609, de 19 de fevereiro de 1998, que trata sobre a Proteção da Propriedade Intelectual de programas de computador, define programa (ou *software*) como:

Art. 1º - Programa de computador é a expressão de um conjunto organizado de instruções em linguagem natural ou codificada, contida em suporte físico de qualquer natureza, de emprego necessário em máquinas automáticas de tratamento da informação, dispositivos, instrumentos ou equipamentos periféricos, baseados em técnica digital ou análoga, para fazê-los funcionar de modo e para fins determinados (BRASIL, 1998, texto digital).

Os programas de computador são algoritmos e processos escritos em uma linguagem de programação e, que após a sua compilação é gerada a linguagem de máquina, que executa determinadas tarefas, conforme especificações anteriormente definidas. Algumas das linguagens de programação mais conhecidas, citadas por Puga e Rissetti (2016), são: *Fortran*, *Cobol*, *Pascal*, *C*, *C++*, *Visual Basic*, *Delphi*, *Java*, entre outras.

Para o desenvolvimento dos programas é necessário que se conheça a lógica envolvida na sua criação e que seja utilizada uma linguagem de programação, que é a representação dos códigos responsáveis por executar os diferentes processos que serão executados pelo programa. Puga e Rissetti (2016, p. 261) descrevem que “a linguagem de programação é formada por palavras”, que são agrupadas em frases com um “determinando significado”, que resolva algum problema proposto. Dessa forma, consideram-se as palavras utilizadas como “palavras-chave” e a frases como “estrutura de programação”.

Puga e Riseti (2016, p. 261) mencionam que:

[...] um programa é constituído de palavras-chave e estruturas de programação definidas segundo as regras (gramática) dessa linguagem, elaboradas de modo que seja mais fácil sua compreensão pelo ser humano. A exemplo da linguagem usada em nossa comunicação, a linguagem de programação possui uma sintaxe, definida por essas regras.

Puga e Riseti (2016) descrevem que seguir esta lógica é importante para que qualquer pessoa que trabalhe com programação possa compreender e fazer alterações, quando necessário. Com isso, escrever o código com uma lógica clara para criar ou interpretar as estruturas dos programas é fundamental para o seu correto funcionamento.

2.2 Algoritmos e Programação

O termo pensamento lógico foi criado por Aristóteles, que na época chamava de “razão”, passando a ser denominado de lógica somente anos mais tarde. A palavra lógica, segundo Puga e Riseti (2016), tem origem do grego “logos” que significa “linguagem racional”. Puga e Riseti (2009, p. 1) apresentam ainda que:

[...] lógica é a análise das formas e leis do pensamento, mas não se preocupa com a produção do pensamento, quer dizer, não se preocupa com o conteúdo do pensamento, mas sim com sua forma, isto é, com a maneira pela qual um pensamento ou ideia é organizada e apresentada, possibilitando que cheguemos a uma conclusão.

O pensamento ou raciocínio lógico é utilizado desde os tempos primitivos, pois o homem precisava estabelecer sequências para a realização de determinadas tarefas, corriqueiras do dia a dia. Isso ocorria de maneira que qualquer pessoa pudesse segui-las e ter sucesso em alcançar as respostas, as quais necessitavam de “coerência e racionalidade” para resolvê-las, conforme descrito por Forbellone e Eberspächer (2005). Este raciocínio é composto por premissas (p) e (q) que constroem uma ideia e levam a uma conclusão (r). Um exemplo é apresentado por Puga e Riseti (2016, p. 1):

p: Sandra é mais velha do que Ricardo.
q: Ricardo é mais velho do que Pedro.
r: Logo, Sandra é mais velha do que Pedro.

É importante salientar que a lógica é comumente associada à matemática, mas vale lembrar que sua aplicabilidade pode ser feita em qualquer área das diferentes ciências, tais como: “a informática, a psicologia, a física e o direito, entre outras” (PUGA; RISSETI, 2009, p. 2).

De acordo com o que foi exposto anteriormente, Forbellone e Eberspächer (2005) enfatizam que a “forma mais complexa do pensamento é o raciocínio” e que a lógica estuda a “correção do raciocínio” e auxilia a colocar em ordem o pensamento, o qual pode ser aplicado para solucionar diferentes tipos de problemas, não só matemáticos. Forbellone e Eberspächer (2005, p. 1-2) ainda descrevem que,

O uso corriqueiro da palavra lógica está normalmente relacionado à coerência e à racionalidade. [...] Podemos ainda dizer que a lógica tem em vista a ‘ordem da razão’. Isso dá a entender que a nossa razão pode funcionar desordenadamente. Por isso, a lógica estuda e ensina a colocar ‘ordem no pensamento’. [...] um dos objetivos da lógica, o estudo de técnicas de formalização, dedução e análise que permitam verificar a validade de argumentos.

Na informática e na computação a lógica é utilizada em todas as áreas, desde a construção das peças (*hardware*) até o desenvolvimento dos programas (*software*). Um exemplo do uso da lógica nessas áreas é apresentado por Puga e Riseti (2016, p. 3), que descrevem a sua utilização no computador:

[...] na construção de um circuito integrado para o teclado (*hardware*), trabalha-se com o conceito de portas lógicas para a verificação da passagem, ou não, de pulsos elétricos de um componente a outro, a fim de que seja estabelecida uma comunicação entre eles. Já no desenvolvimento de *softwares*, é por meio do raciocínio lógico que o programador cria algoritmos que podem ser transformados em programas de computador, capazes de solucionar problemas cada vez mais complexos.

Como visto, a lógica pode ser utilizada em diferentes ciências. Como o foco desta pesquisa é analisar a lógica na criação de algoritmos, a compreensão e utilização desta na programação é fundamental para o desenvolvimento dos códigos e a resolução dos problemas propostos. Consequentemente, criar programas cada vez mais funcionais e que atendam às necessidades das pessoas, das empresas e da sociedade em geral, é algo relevante, conforme descrito por Celestino (2015).

A lógica de programação, segundo Forbellone e Eberspächer (2005, p. 2), está relacionada com a coerência em usar o pensamento para resolver problemas, a partir da utilização de “processos de raciocínio e simbolização formais na

programação de computadores”, e com o desenvolvimento de técnicas para a produção de soluções lógicas que “resolvam com qualidade os problemas que se deseja programar”.

A lógica é importante para quem deseja trabalhar com programação e desenvolvimento de sistemas, pois permite criar a percepção da construção da sequência necessária para a estruturação do código e resolver os problemas apresentados. Moraes (2000, p. 4) define que sequência lógica são os pensamentos que se deve desenvolver para poder descrever “[...] uma sequência de instruções, que devem ser seguidas para se cumprir uma determinada tarefa”. O autor supracitado também descreve que a:

[...] instrução é a informação que indica a um computador uma ação elementar a executar. Convém ressaltar que uma ordem isolada não permite realizar o processo completo, para isso é necessário um conjunto de instruções colocadas em ordem sequencial lógica (MORAES, 2000, p. 4).

Para compreender sobre a utilização da lógica na construção de algoritmos é necessário saber interpretar os problemas que precisam ser resolvidos na programação e, com isso, identificar quais as possibilidades de soluções.

O aprendizado e o aprimoramento da lógica são essenciais para a formação de um bom programador, pois serve de base para o desenvolvimento de sistemas em qualquer linguagem de programação. Forbellone e Eberspächer (2005, p. 2-3) explicam que a lógica de programação “[...] pode ser concebida pela mente treinada e pode ser representada em qualquer uma das inúmeras linguagens de programação existentes”.

Isso se deve ao fato de que as linguagens de programação são diferentes em alguns aspectos, pois foram desenvolvidas com objetivos diversos. Mas com a utilização de algoritmos pode-se apresentar de maneira mais fácil a lógica que se deseja programar, sem que seja necessário especificar qual a linguagem pretendida. De acordo com Forbellone e Eberspächer (2005, p. 3), “o objetivo principal do estudo da Lógica de Programação é a construção de algoritmos coerentes e válidos”. No Quadro 01, Puga e Rissetti (2016, p. 261-262) apresentam uma comparação entre um comando escrito em Algoritmo e em algumas linguagens de programação.

Quadro 01 - Exemplo de algoritmo comparado com linguagem de programação.

<p>Algoritmo - ESCREVA ("Algoritmos e Estruturas de Dados")</p> <p>Pascal - write("Algoritmos e Estruturas de Dados");</p> <p>C - printf("Algoritmos e Estruturas de Dados");</p> <p>C++ - cout<<"Algoritmos e Estruturas de Dados";</p> <p>Java - System.out.println("Algoritmos e Estruturas de Dados");</p>

Fonte: Do Autor, adaptado de Puga e Risettti (2009, p. 261-262).

Na primeira linha do Quadro 01 é apresentado o comando ESCREVA, que é utilizado para representar a instrução que escreverá na tela o termo “Algoritmos e Estruturas de Dados”. A forma de escrita deste comando é bastante parecida com as linguagens de programação apresentadas, conforme as especificidades de cada uma.

Para entender mais sobre os algoritmos, precisa-se saber que a ideia de algoritmo é antiga. Puga e Risettti (2016) apresentam que matemáticos e filósofos já discutiam sobre eles no século XVII, no qual Gottfried von Leibniz foi um proeminente no assunto. Puga e Risettti (2016, p. 8) descrevem que Gottfried já previa “[...] máquinas universais de calcular e estranhas linguagens simbólicas para representar ideias complexas por meio de sinais convencionais”. Nesta época os algoritmos eram utilizados na matemática clássica para substituir “os números por letras” e definir “símbolos que são manipulados por meio de regras práticas”.

Para simplificar, conforme Guedes (2014, p. 3), um algoritmo é uma “sequência de passos que devem ser realizados para alcançar determinado objetivo”, ou seja, um passo a passo dos procedimentos necessários para a resolução de uma tarefa. Nele não é respondida a pergunta ‘o que fazer’, mas sim ‘como fazer’.

Forbellone e Eberspächer (2005, p. 3) descrevem que:

Na medida em que precisamos especificar uma sequência de passos, é necessário utilizar ordem, ou seja, ‘pensar com ordem’, portanto precisamos utilizar lógica. Apesar do nome pouco usual, algoritmos são comuns em nosso cotidiano, como, por exemplo, uma receita de bolo.

Ou seja, uma sequência de processos que podem ser elaborados e seguidos por qualquer pessoa que consiga entender a lógica por trás da estrutura descrita. Além disso, são encontrados, no dia a dia, vários exemplos de algoritmos que precisam ser seguidos para se realizar determinadas tarefas. Como, por exemplo, fazer uma limonada, citado por Guedes (2014, p. 4):

1. escolher os limões;
2. cortar os limões;
3. espremer os limões;
4. adicionar água gelada ao suco do limão;
5. adoçar.

Como no exemplo apresentado, Puga e Rissetti (2016) descrevem que no dia a dia, as pessoas utilizam diferentes tipos de algoritmos de forma intuitiva, sem que percebam e nem precisem planejar o seu processo.

Além dos algoritmos que estão presentes no dia a dia de todos, eles também são utilizados na área de programação, quando é necessário descrever as etapas que serão efetuadas para a execução das tarefas de um programa. Isso facilita a compreensão de quem está no início do aprendizado da programação, pois são normalmente genéricos e consegue-se adaptá-los a qualquer linguagem.

Forbellone e Eberspächer (2005, p. 3) destacam que um “[...] algoritmo tem por objetivo representar mais fielmente o raciocínio envolvido na Lógica de Programação”. Dessa forma, sendo capaz de abstrair a lógica e o processo envolvido na criação de detalhes computacionais que as linguagens de programação requerem (conforme exemplo apresentado no Quadro 01), e isso pode ocasionar uma maior compreensão da resolução dos problemas propostos.

A Figura 02, ilustrada por Guedes (2014, p. 117), mostra o diagrama do processo de criação de um algoritmo em comparação com o desenvolvimento de um programa. Pode-se perceber que os dois processos apresentam alguns passos básicos, inicia com um problema e chega a um resultado final, o “Programa Aplicativo”.

Figura 02 - Diagrama do processo de programação.



Fonte: Guedes (2014, p. 117).

Ao analisar o processo para a escrita do algoritmo, apresentado na Figura 02, parte-se de um problema, sendo preciso analisá-lo para entender o contexto em que este está inserido e, após, pensar as possíveis maneiras de representá-lo, qual o processo e os comandos que podem ser utilizados na criação de uma solução para se chegar a um resultado satisfatório.

Guedes (2014) e Puga e Rissetti (2016) expõem que a melhor maneira para se criar um algoritmo é entender o enunciado do problema, analisar os dados de entrada, pensar como a partir dos dados iniciais chega-se ao resultado final esperado e, somente depois iniciar o processo de escrita do código. Já o processo de escrita de um programa é basicamente o mesmo, mas dependendo do contexto em que este será utilizado pode-se escolher uma linguagem de programação ou outra. Por exemplo, se o sistema será utilizado na *internet*, é necessário utilizar uma linguagem que suporte este tipo meio.

A construção correta de um algoritmo é importante, pois, uma vez proposta uma solução, ela pode ser transcrita para qualquer linguagem de programação, conforme as especificações de cada uma. Na criação de um algoritmo, segundo Forbellone e Eberspächer (2005), é necessário descrever “ações claras e precisas”, e ao partir de um “estado inicial” chega-se a um “estado final e bem definido”. Isso significa que, após a criação de um algoritmo ele deve ser capaz de, ao inserir

dados iniciais, gerar um resultado final, e sempre que forem inseridos os mesmos dados precisa retornar resultados iguais.

Existem basicamente três tipos mais comuns de representação de algoritmos: descrição narrativa, fluxograma e pseudocódigo (PUGA; RISSETTI, 2016; GUEDES, 2014). Ainda, Guedes (2014, p. 7-8) apresenta uma breve descrição de cada tipo:

Descrição narrativa - Consiste em analisar o problema e escrever, utilizando uma linguagem natural, os processos para a sua resolução.

Fluxograma - Consiste em analisar o problema e escrever, utilizando símbolos gráficos predefinidos, os passos para a sua resolução.

Pseudocódigo - Consiste em analisar o problema e escrever, por meio de regras, os passos para sua resolução.

Nesta pesquisa é realizada a análise e o estudo das contribuições do “pseudocódigo”⁴, também chamado de “português estruturado”, que utiliza estruturas de código escritos em Língua Portuguesa para representar os algoritmos na aprendizagem dos estudantes. Este tipo de estrutura é a base utilizada nas aulas das disciplinas de Algoritmos e Programação de Computadores, nas quais os estudantes têm o primeiro contato com a programação.

2.3 Português Estruturado e *VisuAlg*

Uma das formas de representação dos algoritmos é por meio do português estruturado, também conhecido como Pseudocódigo, Portugol ou Pseudolinguagem. Segundo Carvalho (2007, p. 7),

O Português Estruturado na verdade é uma simplificação extrema da língua portuguesa, limitada a pouquíssimas palavras e estruturas que têm significado pré-definido, pois deve-se seguir um padrão. Emprega uma linguagem intermediária entre a linguagem natural e uma linguagem de programação, para descrever os algoritmos.

Além disso, no português estruturado utilizam-se instruções na Língua Portuguesa para representar os comandos, de forma mais restrita do que a linguagem escrita convencional. Cada instrução tem significado bem definido para

⁴ Pseudocódigo - “A palavra *pseudocódigo* significa ‘falso código’. Este nome se deve à proximidade que existe entre um algoritmo escrito em pseudocódigo e a maneira pela qual um programa é representado em uma linguagem de programação” (PUGA; RISSETTI, 2016, p. 12).

identificar os conceitos que se quer representar, na construção do código de um algoritmo (CARVALHO, 2007).

Muitas vezes os professores preferem utilizar o português estruturado para a escrita dos algoritmos, pois representam a sequência de informações que descreve a ordem dos processos e os passos para a resolução de determinado problema. A utilização desta estratégia facilita e simplifica a compreensão do processo, pois está na língua materna dos estudantes em comparação à utilização de uma linguagem de programação, que comumente, é escrita na Língua Inglesa (CARVALHO, 2007).

Berg (1998) e Carvalho (2007) descrevem que a escrita dos algoritmos por meio do português estruturado é bastante simplificada, mas pode representar a maioria dos comandos, testes e laços de repetição que são utilizados nas linguagens de programação, de maneira bem semelhante. O português estruturado tem a característica de ser genérico e permite que seja adaptado a praticamente qualquer linguagem de programação, sendo essa estruturada ou não.

Isso não significa, conforme Carvalho (2007), que resolver os problemas algorítmicos por meio do português estruturado seja uma tarefa simples, mas que não é necessário tanto rigor nos detalhes como nas linguagens de programação. A autora descreve ainda que “[...] o algoritmo não deixa de funcionar porque nos esquecemos de colocar um ‘;’ (ponto e vírgula) por exemplo, já um programa não funcionaria” (CARVALHO, 2007, p. 7).

A Figura 03, apresentada por Carvalho (2007, p. 7), mostra o exemplo de uma sintaxe para a criação de algoritmos com a utilização do português estruturado.

Figura 03 - Exemplo de algoritmo escrito em português estruturado.

```
INICIO
  <instruções>
  SE <teste> ENTAO
    <instruções>
  SENA0
    <instruções>
  FIMSE
FIM
```

Fonte: Carvalho (2007, p. 7).

Ao analisar a Figura 03, percebe-se que a sintaxe da estrutura básica de um algoritmo, na qual são apresentados alguns comandos de exemplo, é de fácil compreensão, pois estão em português e entende-se o significado destes facilmente. Por exemplo, o comando SE...SENAO...FIMSE representa uma estrutura de seleção composta, que após um teste apresenta opções de respostas para SIM, instruções do SE, ou para NÃO, instruções do SENAO. Como a estrutura do algoritmo está na Língua Portuguesa, percebe-se que os estudantes compreendem melhor o significado dos comandos. Com isso, é necessário, principalmente, aprender e aprimorar os conceitos sobre a construção da lógica de programação para a criação da estrutura do algoritmo.

Por meio da experiência empírica do autor desta pesquisa, percebeu-se que alguns professores já utilizam a criação de algoritmos por meio do português estruturado há vários anos, na qual os comandos eram escritos com papel e caneta. Mas, ao perceber as dificuldades que alguns estudantes apresentavam, ao iniciar a aprendizagem de programação com este método de escrita, decidiu-se realizar uma busca para encontrar algum *software* que pudesse auxiliar nesse processo, podendo o estudante visualizar o resultado do algoritmo após a sua construção.

Por meio da busca realizada, constatou-se que foram desenvolvidos vários *softwares* que utilizam o português estruturado para a escrita de algoritmos. Dentre eles Fabri (2008) destaca: Ambiente de Aprendizado de Programação - AMBAP (Universidade Federal de Alagoas), Animação e Simulação de Algoritmos - ASA (Serviço Nacional de Aprendizagem Comercial), G-Portugol (União Pioneira de Integração Social), Portugol Studio (Universidade do Vale do Itajaí), Visualizador de Algoritmos - *VisuAlg* (Apoio Informática Ltda.), dentre outros menos utilizados ou conhecidos.

Para o desenvolvimento da prática pedagógica foi utilizado o *Software VisuAlg*, pois os autores Almeida (2013), Costa (2010), Guedes (2014), Leite et al. (2013) e Souza (2009) descrevem experiências e vantagens que demonstram que este é um *software* muito utilizado, por diferentes professores das disciplinas de Algoritmos e Programação de Computadores em diversas instituições de ensino, principalmente na educação superior. Como citado por Guedes (2014), ele é um “[...] bom recurso para quem está iniciando o aprendizado de algoritmos”. Além disso, o

Software VisuAlg foi escolhido por estar disponível sob a licença de uso *Freeware*⁵, apresentar várias opções para controle e visualização dos dados trabalhados e, também, por não haverem relatos de autores da sua utilização em cursos de nível técnico. Guedes (2014, p. X) ainda menciona que existem “[...] facilidades com a edição direta dos algoritmos, a depuração e o acompanhamento dos valores das variáveis utilizadas, e o acompanhamento de execução passo a passo”.

O *VisuAlg*, abreviação de “Visualizador de Algoritmos”, é um *software* simples de domínio público, com licença *freeware*, que pode ser executado sem a sua instalação, e ocupa aproximadamente 1,2 *megabytes* de espaço em disco. Com isso, possibilita a sua rápida utilização em qualquer computador, tanto com sistema operacional *Windows* ou *Linux*. Foi criado por Cláudio Morgado de Souza, professor da Universidade Severino Sombra - RJ, do Centro de Ensino Superior de Valença - RJ.

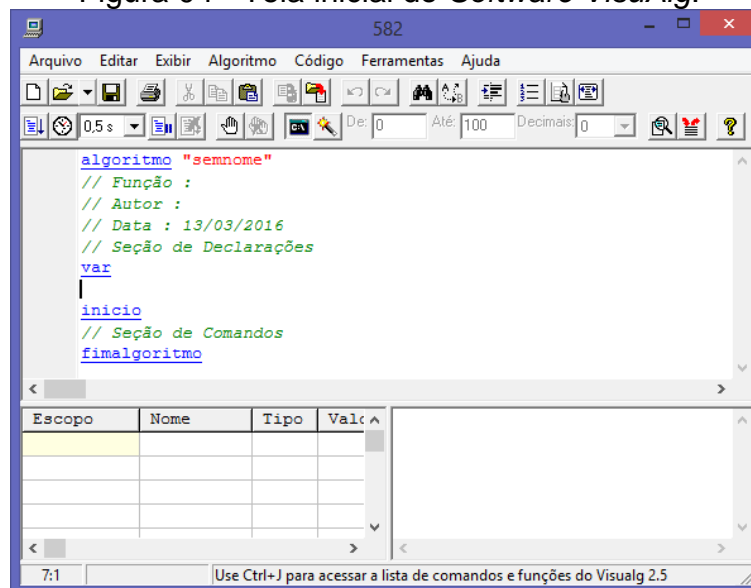
Além do exposto, a empresa Apoio Informática Ltda. (2016, texto digital), que ajudou a desenvolver o *VisuAlg*, menciona que “A linguagem que o *VisuAlg* interpreta é simples: é uma versão portuguesa dos pseudocódigos largamente utilizados nos livros de introdução à programação”. Como descrito no *site* da empresa Apoio Informática Ltda. (2016, texto digital), o *VisuAlg* surgiu da necessidade de “[...] uma ferramenta que permitisse aos alunos iniciantes em programação o exercício dos seus conhecimentos num ambiente próximo da realidade”. Isto é, exercitar a criação de algoritmos em um *software* que tivesse o processo parecido ao de uma linguagem de programação, além de codificar e executar algoritmos em uma linguagem simples, na Língua Portuguesa e com facilidade para edição do que foi escrito.

Por apresentar facilidade para digitar, corrigir, executar e depurar os pseudocódigos, o *VisuAlg* pode ser utilizado por professores nas diversas etapas da construção do conhecimento sobre a lógica de programação, nos diferentes níveis de ensino. Para Guedes (2014), este *software* é considerado “um bom recurso” para ser utilizado no meio acadêmico, não somente para aprender algoritmos, mas

⁵ Freeware - “Programa de domínio público, disponível para qualquer usuário, sem necessidade de pagamento; *software* distribuído livremente” (MICHAELIS, 2009).

também para “melhor entender sua execução”. O uso do *VisuAlg* também é defendido por Costa (2010, p. 6), que cita a existência de “vários recursos didáticos que auxiliam na explicação do funcionamento dos programas”, auxilia na execução do passo a passo do código e na análise do conteúdo das variáveis. A Figura 04 mostra a tela inicial do *Software VisuAlg*.

Figura 04 - Tela inicial do *Software VisuAlg*.



Fonte: Do Autor (2017), a partir do *Software VisuAlg*.

Na Figura 04 constata-se que o *VisuAlg* é um *software* com interface intuitiva de fácil compreensão e utilização, se comparado com as IDEs⁶ disponíveis para as linguagens de programação, mas com funções básicas que auxiliam na construção dos algoritmos. Almeida (2013, p. 2) faz a relação entre o *VisuAlg* e as “[...] rodinhas de apoio que uma criança usa ao aprender a andar de bicicleta, e que são retiradas quando deixam de ser necessárias”.

Para Souza (2009), resolver um problema de lógica por meio da escrita de algoritmos na Língua Portuguesa, com o auxílio do *Software VisuAlg*, facilita a compreensão e aprendizagem dos estudantes sobre o processo de criação da estrutura do código. De acordo com o autor, com o conhecimento dos principais comandos, que são em português, desenvolve-se um código de fácil entendimento. O estudante tem a possibilidade de realizar testes para saber se existem erros de

⁶ IDE - *Integrated Development Environment* (Ambiente de Desenvolvimento Integrado) (PUGA; RISSETTI, 2016, p. 155).

codificação. Com isso, pode-se executar o “programa” criado e visualizar o seu comportamento passo a passo e analisar os dados resultantes.

A Figura 05 mostra um exemplo de algoritmo escrito no *VisuAlg*, no qual foi solicitada a criação de um programa com o seguinte enunciado: Escreva um algoritmo que ao ler a idade de uma pessoa, informa se ela é maior de idade ou não.

Figura 05 - Algoritmo escrito no *Software VisuAlg*.

```

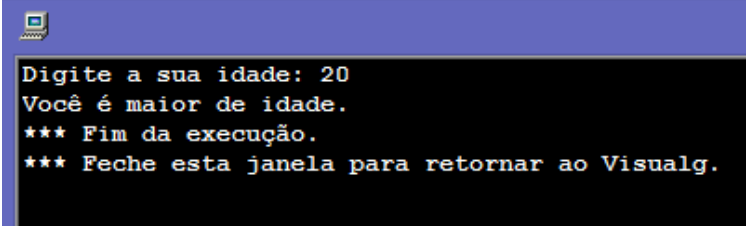
ALGORITMO "exemploVisualg"
VAR
    idade : INTEIRO
INICIO
    ESCREVA ("Digite a sua idade: ")
    LEIA (idade)
    SE (idade >= 18) ENTAO
        ESCREVA ("Você é maior de idade.")
    SENAO
        ESCREVA ("Você é menor de idade.")
    FIMSE
FIMALGORITMO

```

Fonte: Do Autor (2017), a partir do *Software VisuAlg*.

Após a execução do código apresentado na Figura 05, pode-se visualizar na Figura 06, o resultado do teste quando for digitado o número 20 (vinte) na solicitação da idade. O algoritmo escrito solicita que seja digitada uma idade, testa se o número digitado é maior ou igual a 18 (dezoito) e informa se é maior de idade ou não, com base no teste realizado.

Figura 06 - Tela de execução do código da Figura 05.



```

Digite a sua idade: 20
Você é maior de idade.
*** Fim da execução.
*** Feche esta janela para retornar ao Visualg.

```

Fonte: Do Autor (2017), a partir do *Software VisuAlg*.

Para que se possa fazer uma melhor comparação entre a estrutura de criação de um algoritmo com a utilização do *VisuAlg* e o mesmo código escrito no *Dr. Java*⁷,

⁷ Dr. Java - “Dr. Java é um ambiente de desenvolvimento leve para escrever programas em Java. Ele é projetado principalmente para estudantes, fornecendo uma interface intuitiva e a capacidade de avaliar de forma interativa o código Java” (DRJAVA TEAM, 2017, texto digital).

são apresentadas na Figura 07A e na Figura 07B as principais diferenças entre as duas abordagens.

Figura 07 - Comparativo entre Português Estruturado e a Linguagem *Java*.

<pre> ALGORITMO "exemploVisualg" VAR idade : INTEIRO INICIO ESCREVA ("Digite a sua idade: ") LEIA (idade) SE (idade >= 18) ENTAO ESCREVA ("Você é maior de idade.") SENAO ESCREVA ("Você é menor de idade.") FIMSE FIMALGORITMO </pre>	<pre> class exemploJava { public static void main (String arg[]) { int idade; idade = Entrada.leiaInt ("Digite a sua idade: "); if (idade >= 18) System.out.println ("Você é maior de idade."); else System.out.println ("Você é menor de idade."); } } </pre>
(A)	(B)

Fonte: Do Autor (2017). (A) a partir do *Software VisuAlg*. (B) a partir do *Software Dr. Java*.

Observa-se que, tanto na Figura 07A como na Figura 07B, a estrutura do algoritmo é parecida, mas na segunda imagem o estudante precisaria compreender um pouco da Língua Inglesa para saber quais comandos utilizar. É necessário cuidar alguns detalhes que são específicos da linguagem *Java*, como, por exemplo, a utilização de ponto e vírgula (;) ao final de algumas linhas do código, abrir e fechar parênteses e chaves corretamente, letras maiúsculas e minúsculas nos comandos e palavras reservadas do sistema.

Pode-se perceber que a utilização de uma linguagem de programação, como o *Java* apresentado no exemplo da Figura 07B, não é uma função simples, pois é preciso ter atenção a alguns detalhes que são importantes para o seu correto funcionamento, específicos em cada linguagem. Já na utilização do português estruturado, o foco principal não é cuidar a escrita, mas sim a resolução do problema de maneira satisfatória e o desenvolvimento da lógica necessária para resolvê-lo. Mesmo assim é preciso seguir uma lógica para a criação dos algoritmos, de maneira que todos compreendam o que foi escrito. Por isso, é necessário que se utilize de uma estrutura mínima para o seu correto funcionamento.

Vários são os autores e professores que apresentam maneiras e ferramentas que se utilizam do português estruturado, para a representação dos algoritmos em suas aulas e experimentos na área de programação. Além disso, diferentes autores descrevem em seus artigos e publicações as técnicas e *softwares* utilizados como

auxiliares no desenvolvimento da lógica de programação em seus estudantes. As pesquisas apresentadas na próxima seção destacam alguns dos experimentos realizados por eles em suas aulas, descrevem sugestões de como proceder e aprimorar o ensino do algoritmo.

2.4 Trabalhos acerca do ensino de algoritmos

Com o intuito de estabelecer os limites ocupados por esta pesquisa, no contexto de referenciais sobre o estudo de *softwares* que utilizam o português estruturado para a criação de algoritmos, foram selecionados os seguintes artigos que possuem alguma relação com os assuntos: “ensino de algoritmos”, “ensino de programação”, “português estruturado”, “Portugol” ou “*VisuAlg*”. A busca foi realizada nas seguintes bases: Biblioteca Digital da UNIVATES, Base de dados EBSCO, Scielo Brasil, Portal de Periódicos CAPES/MEC e Google Acadêmico.

As buscas realizadas na “Biblioteca Digital da UNIVATES”, “Base de dados EBSCO” e “Scielo Brasil” retornaram artigos e trabalhos que não se relacionavam com o foco desta pesquisa. Os resultados encontrados foram sobre algum tipo de algoritmo ou programa utilizado para testar e validar os resultados de pesquisas em diversas áreas do conhecimento, ou sobre outros meios utilizados para o ensino de algoritmos, não com o uso de um *software*. Por isso, não serão citados, pois pouco poderiam contribuir para o estudo do desenvolvimento da lógica de programação por meio do português estruturado com o *Software VisuAlg*.

Já na busca realizada no “Portal de Periódicos CAPES/MEC” e “Google Acadêmico”, retornaram diversos artigos e dissertações que foram utilizados como referência para o presente estudo. Dentre estes foram selecionados cinco artigos, julgados os mais relevantes para esta pesquisa: “À procura de um contexto para apoiar a aprendizagem inicial de programação” (GOMES; MENDES, 2015), “Auxiliando a aprendizagem de algoritmos com a ferramenta WebPortugol” (HOSTINS; RAABE, 2007), “Ferramentas de apoio ao aprendizado de programação na FAFICA” (SOUZA; FRANÇA; LINS, 2014), “VisuAlg: estudo de caso e análise de

compilador destinado ao ensino de programação” (LEITE et al, 2013) e “*Visualizing the Execution of Programming Worked-out Examples with Portugal*” (MOURA, 2013), que fazem maior relação com a prática pedagógica descrita neste trabalho.

Gomes e Mendes (2015) apresentam em seu artigo a análise de seis estudos que foram realizados em cursos de informática, entre 2006 a 2010, na Universidade de Coimbra, em Portugal, com o intuito de propor mudanças na prática de ensino e, conseqüentemente, na estrutura curricular dos cursos. Os estudos analisados, segundo os autores, foram desenvolvidos por diferentes professores desta instituição, e consistiam em observar quais os motivos que levam os estudantes, que estão no início da aprendizagem da programação, a reprovarem nestas disciplinas. Inicialmente, Gomes e Mendes (2015) apresentam alguns motivos que, segundo eles, são os principais que contribuem para a reprovação dos estudantes, nas disciplinas de algoritmo e programação. Dentre os motivos citados, foi considerado como principal, o obstáculo que é para alguns estudantes saber como iniciar a resolução dos problemas computacionais e perceber os passos para resolvê-los. Este motivo serviu como ponto de partida para os diferentes professores iniciarem seus estudos sobre o desenvolvimento da lógica de programação. Gomes e Mendes (2015, p. 14), baseados nas informações apresentadas pelos professores responsáveis pelos estudos, realizaram a análise destes levando em consideração a ideia de que, “[...] é possível ajudar os estudantes com dificuldades em aprender a programar”. A seguir é realizada uma breve descrição de cada estudo mencionado por Gomes e Mendes (2015), os quais foram denominados em Estudo A, B, C, D, E e F, conforme subdivisão apresentada pelos próprios autores.

O “Estudo A” (*Mathematics and programming problem solving*), realizado em 2006, envolveu estudantes repetentes, na primeira disciplina de programação, que apresentavam “muitas dificuldades de carácter matemático”. Com isso, foi proposto a eles participarem de um “curso especial” sobre resolução de problemas, no qual percebeu-se que apresentavam dificuldades com a matemática, na interpretação dos enunciados, no nível de abstração e no raciocínio lógico. A intenção do curso não era somente “identificar pontos fracos na resolução de problemas, mas também proporcionar aos estudantes alguma formação neste domínio, verificando a sua influência sobre a sua aprendizagem de programação” (GOMES; MENDES, 2015).

No “Estudo B” (*A study on student's characteristics and programming learning*), realizado em 2007 e 2008, foi analisada qual é a influência das “capacidades matemáticas” e de resolução de problemas na aprendizagem da programação. Foi proposta a utilização de duas linguagens de programação diferentes (*C* e *Python*) com dois grupos de estudantes. Neste estudo foram utilizadas “análises quantitativas descritivas”, e foi considerado o “interesse e os resultados obtidos pelos estudantes nas disciplinas introdutórias de programação”. Os professores responsáveis por este estudo perceberam que as dificuldades de aprendizagem continuavam quase que inalteradas em ambas as linguagens. E constataram que para aprender a programar são exigidos “tempo e maturidade” dos estudantes, conforme descrito por Gomes e Mendes (2015).

O “Estudo C” (*A study on basic mathematics knowledge for the enhancement of programming learning skills*), aplicado em 2007 e 2008, foi realizado com seis estudantes da disciplina de Introdução a Programação, que apresentavam dificuldades em aprender a programar. Segundo os autores Gomes e Mendes (2015), o “estudo incluiu duas sessões em que os estudantes foram confrontados com diferentes exercícios, teóricos e práticos [...] em função das dificuldades demonstradas”. O intuito foi de identificar o tipo de dificuldade apresentada pelos estudantes, na matemática ou na programação, e propor atividades para auxiliar na aprendizagem.

No “Estudo D” (*Bloom's taxonomy based approach to learn basic programming*), realizado em 2007 e 2008, foi aplicado aos estudantes um exame baseado na “Taxonomia de Bloom”⁸, no qual havia perguntas com um nível crescente de dificuldade dividida em quatro grupos: conhecimento, compreensão básica, compreensão avançada e aplicação sobre programação. O estudo apresenta que parte dos estudantes obteve resultado satisfatório nas questões do primeiro nível. Já nos outros três níveis, demonstraram baixo desempenho, e pode-se, segundo Gomes e Mendes (2015, p. 19), perceber a “[...] grande lacuna existente

⁸ Taxonomia de Bloom - Auxilia “[...] a identificação e a declaração dos objetivos ligados ao desenvolvimento cognitivo que [...] engloba a aquisição do conhecimento, competência e atitudes, visando facilitar o planejamento do processo de ensino e aprendizagem” (FERRAZ; BELHOT, 2010).

entre a compreensão de código já feito, a construção de pequenas partes de código e a construção integral de um programa”.

O “Estudo E” (*A study on students' behaviors and attitudes towards learning to program*), realizado em 2008 e 2009, com o intuito de analisar se os métodos de estudos eram adequados a aprendizagem de programação. Foi desenvolvido com duas turmas de programação, nas quais foi aplicado um “questionário multidimensional” composto por 44 itens subdivididos em cinco escalas, dentre elas: enfoque compreensivo, enfoque reprodutivo, enfoque percepções pessoais, enfoque motivacional e enfoque organizativo. Conforme Gomes e Mendes (2015, p. 20), os “resultados mostraram que as percepções pessoais e a motivação são fatores importantes e estão relacionados com os resultados nas disciplinas de programação”.

Já no “Estudo F” (*Dificuldades de aprendizagem de programação de computadores: contributos para a sua compreensão e resolução*), realizado em 2008 e 2009, os autores tinham o objetivo de analisar como uma modificação na metodologia de ensino influenciaria no resultado final em uma disciplina de programação. A alteração consistiu em incluir atividades de revisão dos assuntos trabalhados no início das aulas práticas, antes de passar para novos assuntos. Os autores apresentam que os resultados não mostraram diferenças significativas entre os estudantes analisados e os demais estudantes de outras disciplinas.

Pode-se perceber que, em praticamente todos os estudos realizados, os estudantes apresentaram as mesmas dificuldades em relação a programação. Com isso, Gomes e Mendes (2015), propuseram que fossem introduzidas alterações organizacionais na estrutura das disciplinas relacionadas a programação para o ano letivo de 2011/2012. Dentre as modificações estavam: turmas menores, mais atenção às necessidades e dúvidas dos estudantes, atividades individualizadas, aulas teóricas e práticas juntas e com o mesmo professor, contextualização das aulas com o cotidiano e troca da linguagem de programação para outra (*Processing*) mais visual.

Outros autores, Hostins e Raabe (2007), apresentam em seu artigo informações sobre a criação de uma ferramenta (*WebPortugol*) que foi desenvolvida

na Universidade do Vale do Itajaí, em Santa Catarina. Esta ferramenta foi projetada para auxiliar nas aulas iniciais de programação, e deveria apresentar aos estudantes os erros no código escrito junto com dicas de como eles poderiam ser corrigidos. Além da construção da ferramenta foi feita a sua integração com um Sistema Tutor Inteligente que já era utilizado na universidade. A integração do *WebPortugol* com o Sistema Tutor Inteligente serviu para analisar quais eram as atitudes dos estudantes durante a escrita e testes dos algoritmos criados. Com isso, o professor identifica as principais partes do código em que os estudantes tiveram maior dificuldade.

Os testes iniciais constaram que a ferramenta desenvolvida atendia aos requisitos esperados e que pôde-se comprovar que a maioria dos estudantes que verificaram seus códigos, e apresentaram erro, realizaram uma depuração passo a passo após analisar uma possível solução apresentada pelo sistema. Com isso, reforçou-se a crença que os autores tinham de que “este tipo de recurso possa apoiar o desenvolvimento da autonomia e da construção do conhecimento”, (HOSTINS; RAABE, 2007, p. 103).

Um terceiro estudo desenvolvido por Souza, França e Lins (2014) na Faculdade de Filosofia Ciências e Letras de Caruaru, em Pernambuco, expõe uma análise das principais dificuldades apresentadas pelos estudantes nas disciplinas de programação, nos diferentes períodos e níveis de aprofundamento de exigência requeridos no desenvolvimento de sistemas computacionais. A ideia da realização deste estudo surgiu a partir de pesquisas que apresentam a falta de motivação e a dificuldade de aprendizagem da lógica de programação, em estudantes que estão no início de cursos de computação.

Segundo Souza, França e Lins (2014, p. 1576), a “forma como o aprendizado de programação é desenvolvido implica na competência do programador profissional”. Baseados nos fatores analisados, os autores propuseram a análise das funcionalidades de nove ferramentas (*CodeMonster*, *FutCode*, *Guido*, *VanRobot*, *KidsRuby*, *RobotProg*, *TBC-AED*, *VisuAlg* e *WebPortugol*) que poderiam auxiliar na aprendizagem da programação nos quatro períodos (semestres), no qual eram estudadas estas disciplinas, no curso Análise e Desenvolvimento de Sistemas.

Para analisar qual ferramenta seria capaz de contribuir em cada período, foi aplicado, aos estudantes, um questionário que analisou três áreas: dados demográficos, estudo de programação e autoavaliação. A primeira parte do questionário solicitava o local onde os estudantes residiam, pois, o cansaço da viagem poderia ser considerado um motivo de desmotivação nas aulas. A segunda parte visava avaliar o interesse em relação aos estudos e conhecer as principais dificuldades no aprendizado em programação. Por fim, na terceira parte, era proposta uma autoavaliação sobre vários conceitos utilizados na programação.

Após a análise das respostas dos questionários, os autores identificaram as necessidades dos estudantes e classificaram a utilização das ferramentas propostas em três níveis de complexidade: iniciante, intermediário e avançado. Aos iniciantes foi sugerido a utilização das ferramentas *RobotProg* e *Guido VanRobot*, para desenvolver os conceitos básicos (operações aritméticas e lógicas, estrutura de controle e construção de algoritmos). Aos do nível intermediário o *Code Monster* e *KidsRuby*, para trabalhar com conceitos de alocação de memória em variáveis e vetores, tipo abstrato de dados e funções. Já aos do nível avançado o *FutCode* para que os estudantes pudessem visualizar a solução de problemas em domínios reais (conceitos avançados, como compilação e recursão). Outras ferramentas poderiam ser utilizadas, mas os autores propuseram a utilização das que mais possibilitariam trabalhar os conceitos específicos de cada nível.

Leite et al. (2013), em outro estudo, apresentam uma pesquisa sobre a utilização e satisfação dos estudantes em relação ao uso do *Software VisuAlg*, no curso de Ciência da Computação, da Universidade Estadual de Londrina, no Paraná. Inicialmente descrevem que o uso de *softwares* na educação tem auxiliado no processo de ensino e de aprendizagem dos estudantes, pois permitem a melhor “[...] fixação dos conceitos e práticas” (LEITE et al., 2013, p. 637), por meio do aprendizado mais “prazeroso, espontâneo e criativo”. Dentre as disciplinas, em que pode ser utilizado algum *software* como recurso didático, foi selecionada a Introdução a Programação, pois nesta, muitos estudantes se decepcionam e perdem a motivação devido à exigência de habilidade de abstração e do raciocínio lógico.

Dentre as ferramentas disponíveis, foi selecionado o *VisuAlg*, pois apresenta disponibilidade de literatura na área e a possibilidade de seu uso para testes e

discussões. Os autores destacaram, também, que nas próximas etapas da pesquisa seriam utilizados outros *softwares* similares. Algumas vantagens apresentadas pela escolha do *VisuAlg* foram a de que a sua linguagem é clara e de fácil compreensão, e que com os testes nos algoritmos visualiza-se de maneira rápida e direta seus resultados. Foi destacado, também, que a ferramenta em questão apresenta desvantagens, como de não aceitar comandos em uma mesma linha e pode gerar confusão na implementação do laço de repetição PARA.

Algumas das questões analisadas sobre o uso do *VisuAlg* nas aulas foram, segundo Leite et al. (2013, p. 639): “*design* da interface, interação com o usuário, facilidade na aplicação dos conceitos teóricos, confiabilidade e a facilidade no aprendizado do conteúdo”. No geral, na média final da avaliação, o *software* obteve uma nota 7.7, pois o processo de aprendizagem dos algoritmos foi agilizado e alcançou os objetivos esperados pelo professor da disciplina. A média só não foi maior porque estudantes que já tinham conhecimento de programação avaliaram o *VisuAlg* como não muito confiável, por causa das desvantagens apresentadas anteriormente.

Por fim, o artigo de Moura (2013) descreve o experimento realizado no ano de 2011 na disciplina de programação da graduação em Engenharia de Polímeros integrado ao Mestrado, da Universidade de Minho, em Portugal, e apresenta uma comparação com os resultados apresentados pelos estudantes no ano de 2010 nesta mesma disciplina.

Moura (2013) apresenta que o ensino da programação, para os estudantes iniciantes em disciplinas relacionadas a informática e computação, é considerado um assunto “pesado” e de difícil compreensão. Com isso, no ano de 2010, foi proposto que o foco nas aulas iniciais de programação fosse ensinar as construções básicas por meio de algoritmos prontos que os estudantes deveriam analisar, testar e depurá-los para compreender como este funciona. O exemplo inicial era completo e sem erros, mas com o aprofundamento dos conteúdos o professor apresentava novos algoritmos, com cada vez mais erros e omissões de códigos para que os estudantes pudessem realizar as correções necessárias até não apresentar mais erros de compilação. Já na turma de 2011, o autor propôs a utilização do *Software Portugal IDE 2.3* como ferramenta para auxiliar no teste e construção dos

algoritmos. O objetivo inicial era os estudantes utilizarem a ferramenta para conferir o funcionamento do código por meio da execução passo a passo deste, conferir e corrigir os erros apresentados, até conseguirem desenvolver um algoritmo completo.

Ao comparar os resultados obtidos nas turmas de 2010 e 2011, o autor descreve que a utilização de uma ferramenta, para visualizar o funcionamento do algoritmo, aumenta o envolvimento, a compreensão e auxilia nos processos mentais necessários para a construção da lógica de programação.

Percebe-se que, ao analisar os trabalhos e pesquisas apresentados anteriormente, as dificuldades na aprendizagem de algoritmos e programação são comuns nos diferentes níveis da educação e em várias instituições de ensino. Por isso, professores vêm propondo e utilizando diversos métodos e ferramentas, que podem auxiliar no desenvolvimento da lógica necessária para a construção de programas que possam resolver os problemas apresentados de maneira eficiente e corretos.

Dessa forma, os estudos apresentados sobre a utilização de diferentes ferramentas para auxiliar no ensino de programação, vêm complementar e dar suporte teórico à prática pedagógica desenvolvida. Em síntese, eles mostram que as dificuldades apresentadas pelos estudantes, que estão no início de disciplinas relacionadas à programação, são comuns a várias instituições de ensino em diversos estados e em outros países. Os estudos ilustram, também, que os professores já percebem essas dificuldades há algum tempo, e apresentam sugestões para tentar saná-las com a proposta de modificar a estrutura das aulas, as bases curriculares e, também, utilizar diferentes ferramentas para dar suporte ao ensino de programação. Dentre essas ferramentas estão os *softwares* que auxiliam no ensino de algoritmos, que utilizam os códigos em português, nos quais os estudantes podem escrever os programas, testá-los e analisá-los passo a passo para encontrar onde estão os erros e inconsistências.

3 PROCEDIMENTOS METODOLÓGICOS

Neste capítulo é apresentada a abordagem metodológica que embasa esta pesquisa, bem como o local no qual foi desenvolvida a prática pedagógica, as atividades, os *softwares*, os materiais que foram utilizados, os procedimentos, os instrumentos para a coleta dos dados e a caracterização da turma. A pesquisa aqui relatada é de cunho qualitativo, descrita como tendo aproximações com estudo de caso, pois será analisada a prática pedagógica desenvolvida com os sete estudantes, da disciplina em que foi discutida a escrita de algoritmos.

3.1 Caracterização da pesquisa

Ao analisar a questão de pesquisa e os objetivos propostos, foi definido que a natureza desta pesquisa é qualitativa. Com a pesquisa qualitativa, Richardson (1999) menciona que, pode-se interpretar os dados coletados de forma a compreender e vivenciar as situações analisadas, os processos desenvolvidos com o grupo de estudantes foco da prática pedagógica e não se limitar às informações genéricas de algumas perguntas estruturadas. Moreira (2011, p. 51) afirma que,

O pesquisador enriquece sua narrativa com trechos de entrevistas, excertos de suas anotações, vinhetas, exemplos de trabalhos de alunos, entremeados de comentários interpretativos procurando persuadir o leitor, buscando apresentar evidências que suportem sua interpretação e, ao mesmo tempo, permitem ao leitor fazer julgamentos de modo a concordar ou não com as asserções interpretativas do pesquisador.

Os dados foram coletados por meio do questionário de sondagem (APÊNDICE D), aplicado no início da prática pedagógica, da prática pedagógica (APÊNDICES E, F, G, H, I, J, K, L, M e N) e pelo questionário de avaliação (APÊNDICE O) efetivado ao final da prática.

Com o questionário de sondagem verificou-se os conhecimentos dos estudantes em relação à programação. Com isso, pôde-se adaptar os assuntos, explicações e exercícios conforme os saberes já adquiridos pelos estudantes. Ao final da prática foram analisadas as atividades desenvolvidas, com o *Software VisuAlg*, para saber como estas contribuíram para o ensino de algoritmos. Para isso, foi realizada a análise dos exercícios efetuados pelos estudantes, dos questionamentos levantados durante a prática pedagógica e do questionário de avaliação da prática (APÊNDICE O).

Além do exposto, durante o desenvolvimento da prática pedagógica observou-se os materiais coletados, realizando-se uma análise preliminar, à luz da fundamentação teórica já apresentada acerca dos principais assuntos que estavam sendo trabalhados sobre programação. E, conforme surgiram as dúvidas, comentários e contribuições dos estudantes, o foco das aulas e atividades propostas puderam ser revistas de maneira a contemplar algum assunto não compreendido por todos os participantes. Com isso, consideraram-se os conhecimentos já construídos anteriormente pelos estudantes (analisados por meio do questionário apresentado no APÊNDICE D), sobre programação, como ponto de partida para o desenvolvimento as aulas.

Para melhor organizar os dados qualitativos, coletados no decorrer da pesquisa, realizou-se anotações em um diário de campo, no qual registrou-se o desenvolvimento das aulas, as atividades, os acontecimentos ocorridos durante a prática, bem como as falas de alguns estudantes, além do registro por fotos, gravação de áudio e os códigos dos algoritmos criados para a resolução dos exercícios propostos.

Segundo Martins (2008), o diário de campo é um documento no qual pode-se registrar os detalhes dos acontecimentos e atividades desenvolvidas durante a pesquisa. Martins (2008, p. 10) apresenta que nos estudos de caso é importante

registrar as informações, organizar rascunhos, transcrições, registros de comentários e opiniões em um “diário de campo, ou diário da pesquisa”.

Na seção seguinte é apresentada a organização da pesquisa, na qual são descritas as etapas que foram realizadas na prática pedagógica.

3.2 Organização da Pesquisa

A presente pesquisa resultou da prática pedagógica desenvolvida e apresenta características de um estudo de caso, na qual faz-se necessária uma análise mais aprofundada dos dados coletados. O estudo de caso é um método de pesquisa que pode ser utilizado, conforme mencionado por Yin (2015, p. 4), “[...] em muitas situações, para contribuir ao nosso conhecimento dos fenômenos individuais, grupais, organizacionais, sociais, políticos e relacionados”, e permite que o pesquisador possa observar características de “eventos da vida real”. Além disso, o estudo de caso ajuda a entender fenômenos sociais complexos e

[...] permite que os investigadores foquem um “caso” e retenham uma perspectiva holística e do mundo real - como no estudo dos ciclos individuais da vida, o comportamento dos pequenos grupos, os processos organizacionais e administrativos, a mudança de vizinhança, o desempenho escolar, as relações internacionais e a maturação das indústrias (YIN, 2015, p. 4).

São detalhados, ainda, aspectos relevantes sobre a importância da construção da lógica de programação, por meio de algoritmos escritos em português estruturado.

A prática pedagógica foi realizada com os sete estudantes da disciplina de Algoritmos do Módulo III - Assistente em Programação, do curso Técnico em Informática da Escola Estadual de Educação Profissional Estrela, localizada no município de Estrela/RS. As aulas desta disciplina ocorreram em três dias da semana consecutivos (terça, quarta e quinta-feira), no turno da noite, cada encontro com a duração de quatro horas-aula. Como a prática proposta era de vinte horas, foram necessárias três semanas para a conclusão desta, pois na primeira semana a prática iniciou na quinta-feira.

3.3 Instrumentos utilizados para coleta de dados

A presente pesquisa foi organizada em cinco momentos, sendo eles: conversa inicial com a direção da escola e assinatura do Termo de Concordância da Direção da Instituição de Ensino (APÊNDICE A); assinatura da Carta de Anuência (APÊNDICE B) autorizando a utilização do nome da escola nos documentos relacionados a esta pesquisa; apresentação da proposta da prática pedagógica aos estudantes da turma foco desta pesquisa e assinatura do Termo de Consentimento Livre Esclarecido (APÊNDICE C); aplicação do questionário de sondagem sobre conhecimentos em relação à programação (APÊNDICE D); desenvolvimento da prática pedagógica (APÊNDICES E, F, G, H, I, J, K, L, M e N); e aplicação do questionário de avaliação da prática pedagógica (APÊNDICE O). Nos próximos itens são descritos cada momento detalhadamente.

1º) Termo de Concordância da Direção da Instituição de Ensino (APÊNDICE A)

Primeiramente foi agendada uma reunião com a direção e coordenação pedagógica para apresentar a proposta da prática e conferir a possibilidade da aplicação desta em uma turma da escola. Tanto a diretora como a coordenadora pedagógica mostraram interesse pela prática, julgando bastante relevante a temática proposta por esta pesquisa. Durante a reunião ficou decidido que a prática seria realizada na turma que iniciou no segundo semestre do ano de 2016, na disciplina inicial do módulo de programação.

Com o aval da equipe diretiva realizou-se uma conversa com o professor de programação responsável pela disciplina de Algoritmos, para serem combinados os detalhes sobre como estava estruturada a pesquisa. O autor desta pesquisa foi professor do referido estabelecimento de ensino, já conhecendo grande parte do corpo docente da escola, estando familiarizado com os conteúdos e dificuldades dos estudantes na aprendizagem de programação. Cabe destacar que no semestre da aplicação da prática pedagógica o pesquisador não era o professor titular da turma.

Assim, na semana seguinte, foi entregue a diretora o Termo de Concordância da Direção da Instituição de Ensino (APÊNDICE A) para que fosse assinado. Nesta mesma oportunidade a diretora reiterou estar satisfeita com a realização da pesquisa e solicitou que gostaria que o nome da escola fosse divulgado nos documentos escritos sobre a prática pedagógica. Ela comentou que a utilização do nome da escola poderia dar maior destaque ao estabelecimento de ensino, divulgando ainda mais os seus trabalhos e projetos pedagógicos. Com isso, foi solicitada a direção que uma Carta de Anuência (APÊNDICE B) fosse redigida, para informar que estavam de acordo com os termos da pesquisa e autorizar a utilização do nome da instituição nesta dissertação e publicações futuras.

2º) Termo de Consentimento Livre Esclarecido (APÊNDICE C)

Recebido a concordância e anuência da direção e coordenação pedagógica, foi reiterado com o professor de programação sobre as melhores datas e os assuntos que seriam trabalhados, na aplicação da prática com a turma selecionada. Após todas as datas e detalhes terem sido acertados com o professor titular, o pesquisador foi encaminhado à turma para iniciar a prática.

No primeiro encontro, os estudantes foram informados sobre a realização da prática pedagógica, foi realizada a apresentação do pesquisador, dos objetivos, dos assuntos e das atividades para serem desenvolvidos, juntamente com os recursos e meios que seriam utilizados. Foi comunicado aos estudantes que eles poderiam optar em participar ou não da investigação. Caso aceitassem era necessário assinar o Termo de Consentimento Livre Esclarecido (APÊNDICE C), concordando com a participação na pesquisa. Todos os estudantes da turma se mostraram interessados na prática e assinaram o termo. Dentre os estudantes somente um era menor de idade, sendo necessário que seus pais o autorizassem a participar da prática pedagógica. Com isso, o estudante levou o termo para casa trazendo-o assinado na aula seguinte.

3º) Questionário de sondagem (APÊNDICE D)

Após a assinatura do Termo de Consentimento Livre Esclarecido, foi solicitado que todos os estudantes respondessem a um questionário de sondagem sobre conhecimentos em relação à programação (APÊNDICE D). O objetivo do questionário foi verificar se os estudantes já tinham algum conhecimento sobre lógica, algoritmo e programação.

Verificar os conhecimentos sobre algoritmos auxiliou a conhecer melhor os estudantes e, também, na organização, no planejamento das aulas e na construção das atividades que foram desenvolvidas durante a prática. A partir da análise das respostas apresentadas, percebeu-se que grande parte dos estudantes não conhecia a programação e nem a criação de algoritmos. As exceções foram o Estudante A⁹ que apresentou algum conhecimento básico sobre a estrutura e criação do algoritmo. Este estudante comentou que o pai já trabalhou com programação na Linguagem C, com isso, conhecia um pouco sobre o assunto, bem como o Estudante E que trabalha há aproximadamente um ano em uma empresa de desenvolvimento de *software*.

A aplicação do questionário de sondagem, antes do início da prática pedagógica, serviu como base para a estruturação e organização dos assuntos que foram trabalhados. Com isso, pôde-se planejar como seriam as próximas aulas, baseando-se nos conhecimentos já adquiridos anteriormente pelos estudantes.

4º) Prática Pedagógica

Várias são as tentativas de fazer com que os estudantes compreendam melhor a lógica de programação trabalhada em disciplinas como Algoritmos e

⁹ Para garantir o anonimato e a integridade dos sujeitos envolvidos na prática pedagógica, no decorrer das análises os estudantes não foram identificados pelo nome, mas sim com letras: Estudante A, Estudante B e assim sucessivamente.

Programação de Computadores (seção 2.4). Em relação à Informática e Computação, podemos inferir que os professores se utilizam de diferentes estratégias e recursos didáticos para auxiliar nos processos de ensino e de aprendizagem desta lógica, tanto nos cursos técnicos e como no ensino superior.

Para esta prática pedagógica (APÊNDICES E, F, G, H, I, J, K, L, M e N) foram desenvolvidos cinco encontros de quatro horas-aula cada, num total de vinte horas-aula. Os assuntos que foram trabalhados podem ser visualizados no Quadro 01 (seção 3.4), que apresenta os conteúdos, as atividades, os recursos, os objetivos, as práticas, os meios de avaliação e os recursos utilizados.

5º) Questionário de avaliação da prática pedagógica

A aplicação do questionário de avaliação da prática pedagógica (APÊNDICE O) teve o intuito de saber quais os pontos fortes e a melhorar da utilização do *Software VisuAlg* como recurso didático nas aulas de início da programação, sob o ponto de vista dos estudantes. E, também, analisar as impressões dos estudantes em participar da prática proposta.

3.4 Estrutura da prática pedagógica

A prática pedagógica foi desenvolvida tendo-se como base as atividades descritas no Quadro 02, que estão mais detalhadas nos Apêndices E, F, G, H, I, J, K, L, M, N.

Quadro 02 - Atividades realizadas durante a prática pedagógica.

AULA	CONTEÚDOS	RECURSOS	OBJETIVOS
01	<ul style="list-style-type: none"> - Proposta da prática pedagógica. - Termo de Consentimento Livre Esclarecido (APÊNDICE C). - Questionário de sondagem (APÊNDICE D). 	<ul style="list-style-type: none"> - Caderno e lápis ou editor de texto. - Lista de exercícios (APÊNDICE J). - Laboratório de Informática. 	<ul style="list-style-type: none"> - Entender o funcionamento da pesquisa. - Assinar o Termo de Consentimento Livre Esclarecido. - Conhecer o conceito de

	<ul style="list-style-type: none"> - Apresentação geral dos assuntos que seriam trabalhados durante a prática. - Conceito de problema, lógica e algoritmo. - Operadores aritméticos. - Tipos de dados primitivos. - Conceito de variáveis e constantes. - Estrutura do algoritmo. 	<ul style="list-style-type: none"> - Projetor multimídia. - Quadro branco. 	<p>problema e lógica.</p> <ul style="list-style-type: none"> - Conhecer os principais operadores aritméticos. - Conhecer e saber identificar os tipos de dados primitivos. - Conhecer conceitos de variáveis e constantes. - Conhecer e saber construir a estrutura de um algoritmo (descrição narrativa).
02	<ul style="list-style-type: none"> - Operadores relacionais, lógicos e de atribuição. - Tipos de representações de algoritmos. - Conceito pseudocódigo e português estruturado. - Criação de algoritmos em português estruturado. - <i>Software VisuAlg</i>. 	<ul style="list-style-type: none"> - Caderno e lápis ou editor de texto. - Lista de exercícios (APÊNDICE J). - Laboratório de Informática. - Projetor multimídia. - Quadro branco. 	<ul style="list-style-type: none"> - Conhecer e saber aplicar os principais operadores relacionais, lógicos e de atribuição. - Conhecer os tipos de representações de algoritmos. - Conhecer o conceito de pseudocódigo e português estruturado. - Conhecer a estrutura para a criação de um algoritmo, em português estruturado, com a utilização do <i>Software VisuAlg</i>.
03	<ul style="list-style-type: none"> - Comandos de entrada e saída de dados. - Comando condicional SE (simples, composto e encadeado). - Utilização do <i>Software VisuAlg</i> para elaborar e testar os algoritmos desenvolvidos. 	<ul style="list-style-type: none"> - Lista de exercícios práticos (APÊNDICE K). - Manual do <i>VisuAlg</i>. - Laboratório de informática. - Projetor multimídia. - Quadro branco. - <i>Software VisuAlg</i>. 	<ul style="list-style-type: none"> - Conhecer os comandos de entrada e saída de dados. - Conhecer o comando condicional SE (simples, composto e encadeado). - Conhecer a utilização e opções do <i>Software VisuAlg</i> para elaborar e testar os algoritmos desenvolvidos. - Saber utilizar o <i>Software VisuAlg</i>.
04	<ul style="list-style-type: none"> - Discussão sobre a utilização do <i>Software VisuAlg</i>. - Comando de repetição PARA. 	<ul style="list-style-type: none"> - Lista de exercícios práticos (APÊNDICE L). - Laboratório de informática. - Projetor multimídia. - Quadro branco. - <i>Software VisuAlg</i>. 	<ul style="list-style-type: none"> - Conhecer o comando para repetição de código (PARA).
05	<ul style="list-style-type: none"> - Comando de repetição ENQUANTO. - Discussão sobre as aulas e proposta pedagógica. - Discussão sobre a escrita de algoritmos no caderno (editor de texto) e com o uso do <i>Software VisuAlg</i>. - Aplicação do questionário de avaliação das aulas (APÊNDICE N). 	<ul style="list-style-type: none"> - Lista de exercícios práticos (APÊNDICE M). - Laboratório de informática. - Projetor multimídia. - Quadro branco. - <i>Software VisuAlg</i>. 	<ul style="list-style-type: none"> - Conhecer o comando para repetição de código (ENQUANTO). - Investigar a percepção dos estudantes sobre prática desenvolvida e os conhecimentos sobre algoritmos adquiridos.

3.5 Curso Técnico em Informática da Escola Estadual de Educação Profissional Estrela

A prática pedagógica foi realizada na disciplina de Algoritmos do Módulo III - Assistente em Programação, do curso Técnico em Informática da Escola Estadual de Educação Profissional Estrela, no município de Estrela/RS.

Para melhor atender o público interessado e organizar os assuntos a serem estudados, o curso Técnico em Informática tem foco em três áreas de qualificações profissionais distintas. Ele está dividido em três módulos: Módulo I - Assistente em Manutenção de Computadores, Módulo II - Assistente em Instalação e Configuração de Redes de Computadores e Módulo III - Assistente em Programação. Ao final de cada módulo o estudante recebe um certificado de que está qualificado a desempenhar as atribuições relativas a cada etapa de estudo. Ao concluir os três módulos, o estudante recebe o diploma de Técnico em Informática, e está apto a ingressar no mercado de trabalho (EEEPE, 2015).

Com o propósito de aperfeiçoar o conhecimento dos estudantes e melhor qualificá-los, a direção e coordenação pedagógica da escola incentivam a participação em diferentes eventos e feiras, tais como, Mostra de Ensino, Extensão e Pesquisa da UNIVATES (MEEP), Mostra das Escolas de Educação Profissional, Ensino Médio Politécnico e Curso Normal (MEP), Feira Estadual de Ciência e Tecnologia da Educação Profissional (FECITEP), Mostra Brasileira de Ciência e Tecnologia e Mostra Internacional de Ciência e Tecnologia (MOSTRATEC), *Proyecto Multimedia - Concurso Latinoamericano de Proyectos de Cómputo* (México), além de diferentes visitas técnicas consideradas importantes pelo corpo docente.

As participações em diferentes atividades extraclasse vêm incentivar e auxiliar no ensino e na aprendizagem dos estudantes, pois pode-se visualizar e desenvolver os conteúdos trabalhados em sala de aula, visando a integração entre a teoria e a prática.

Vale ressaltar que, conforme descrito no Projeto Político Pedagógico da EEEPE (2015), a instituição faz parte da rede pública estadual de ensino, sendo

umas das cinco escolas somente técnicas mantidas pelo Governo do Estado do Rio Grande do Sul. Ela conta ainda com os cursos técnicos em Confeitaria, Edificações, Recursos Humanos e Secretariado, tendo o seu funcionamento nos turnos tarde e noite.

3.5.1 Caracterização da turma

A prática pedagógica foi realizada com o Módulo III - Assistente em Programação, do primeiro semestre de um total de dois que compõem esta etapa final do curso Técnico em Informática. A turma era composta por sete estudantes do sexo masculino, com idades entre 17 a 22 anos, os quais são oriundos de diversas escolas da região do Vale do Taquari, que estão cursando ou já concluíram o Ensino Médio. Segundo o professor titular da disciplina de Algoritmos, estes sete estudantes iniciaram o curso juntos, sempre na mesma turma. Com isso, acredita-se que teve um maior entrosamento entre eles durante a resolução das atividades e exercícios propostos.

Estes estudantes procuraram o curso para conhecerem mais sobre a área de informática, se atualizar sobre assuntos de tecnologia ou obterem uma certificação em nível técnico para tentarem se inserir no mercado de trabalho. Além destes, procuram também o curso, profissionais de empresas para aperfeiçoarem seus conhecimentos em relação à área técnica de informática (EEEPE, 2015).

3.6 Categorização dos dados

Os dados coletados nesta pesquisa foram divididos em categorias, analisados e discutidos com aproximações à Análise Textual Discursiva, conforme propõem Moraes e Galiazzi (2013). Os autores apresentam que pode-se organizar os dados de maneira que estes possam ser melhor analisados seguindo alguns passos como: fragmentação e análise do material coletado com detalhes; estabelecer relações ou

categorização, agrupando os dados correlatos; captação do emergente, observação e análise do que surgiu de novo durante o processo; e processo de auto-organização, deixar que novas ideias sejam inseridas na compreensão do todo.

Moraes e Galianzi (2013) descrevem que é necessário ter “presente a relação entre leitura e interpretação” ao iniciar a análise qualitativa. Estes autores referem-se a Análise Textual Discursiva como “[...] uma metodologia de análise de dados e informações de natureza qualitativa com a finalidade de produzir novas compreensões sobre os fenômenos e discursos” (MORAES; GALIANZI, 2013, p. 7). Isso deve-se ao fato de que “todo texto possibilita uma multiplicidade de leituras”, relacionadas as intenções e entendimentos do autor e dos leitores, baseando-se em “alguma perspectiva teórica, seja esta consciente ou não” (MORAES; GALIANZI, 2013, p. 13).

Além disso, é fundamental ser delimitado o *corpus* da análise textual, isso é, os materiais coletados podem ter sido produzidos para a pesquisa, como: entrevistas, registros de observação, depoimentos e diários diversos (MORAES; GALIANZI, 2013). Os autores citados descrevem o processo da análise textual discursiva em três etapas:

- *unitarização*, nesta etapa ocorre a “desmontagem dos textos”, que consiste em explorar os dados em busca de proposições e conceitos relativos ao estudo em questão;
- *categorização*, consiste em “[...] construir relações entre as unidades de base, combinando-as e classificando-as, reunindo esses elementos unitários na formação de conjuntos que congregam elementos próximos, resultando daí sistemas de categorias” (MORAES; GALIANZI, 2013, p. 12);
- *construção do metatexto*, que consiste na elaboração de novos textos que representem o conjunto de informações coletadas, constituídos de “[...] descrição e interpretação, representando o conjunto um modo de teorização sobre os fenômenos investigados” (MORAES; GALIANZI, 2013, p. 32).

Diante disso, os registros no diário de campo, as gravações e os códigos salvos dos estudantes durante a prática pedagógica, foram agrupados conforme as

suas características em comum. Para Moraes e Galiuzzi (2013, p. 22), a categorização “[...] é um processo de comparação constante entre as unidades definidas no momento inicial da análise, levando a agrupamentos de elementos semelhantes”. A relação entre os dados coletados foram se estabelecendo a partir da escuta e transcrição dos áudios gravados durante a prática, análise dos algoritmos desenvolvidos pelos estudantes e percepções anotadas no diário de campo.

No próximo capítulo, é apresentada a análise realizada com base nos dados coletados durante a prática pedagógica e os resultados obtidos na pesquisa.

4 ANÁLISE E RESULTADOS

Neste capítulo é realizada a análise dos dados coletados na prática pedagógica desenvolvida com os estudantes do Módulo III - Assistente em Programação, do curso Técnico em Informática. Está dividido em três seções, que surgiram a partir dos objetivos específicos: 1 - Conhecimentos sobre algoritmos; 2 - Contribuições do *Software VisuAlg* no ensino de algoritmos; 3 - Percepções dos estudantes sobre o *Software VisuAlg*.

É preciso destacar que, na primeira categoria, são descritos os conhecimentos sobre algoritmos dos estudantes, após a análise das respostas apresentadas no questionário de sondagem (APÊNDICE D). Na segunda categoria é realizada a análise da prática pedagógica, buscando compreender como os estudantes fazem a relação entre a lógica de programação apresentada e a construção dos algoritmos em português estruturado. Já na terceira categoria é apresentada a análise de como o *Software VisuAlg* auxiliou na construção da lógica necessária para o desenvolvimento de algoritmos, com base nas respostas apresentadas no questionário de avaliação da prática (APÊNDICE O).

4.1 Categoria 1 - Conhecimentos sobre algoritmos

Para atingir o primeiro objetivo específico “Identificar os conhecimentos dos estudantes em relação à programação”, os estudantes responderam ao questionário

de sondagem (APÊNDICE D), para perceber quais os conhecimentos estes apresentavam sobre lógica, algoritmos e programação.

Saber os conhecimentos dos estudantes sobre algoritmos é importante, pois pode-se organizar e pensar melhor a proposta da prática pedagógica, partindo-se do que os estudantes sabem, ou que não conhecem, para propor novas explicações e atividades. Para Moreira (2011), fazer a ligação entre os conhecimentos já adquiridos e os novos, a serem trabalhados, é fundamental para os processos de aprendizagem, pois os envolvidos podem criar sentido e fazer a ligação entre o que sabem e o que está se aprendendo, dando novos significados aos conhecimentos já adquiridos.

O questionário continha nove questões, sendo que as três primeiras foram mais gerais para conhecer um pouco sobre os estudantes, suas idades, formação escolar e se já trabalhavam com informática. Com a análise destas questões percebe-se que a faixa etária dos estudantes está entre 17 e 22 anos de idade, que dos sete estudantes, dois ainda estavam cursando o Ensino Médio e somente dois trabalham com informática. Um destes, o Estudante E, trabalha há aproximadamente um ano em uma empresa de desenvolvimento de *software*, mas não atua diretamente na programação.

As outras questões estavam relacionadas ao conhecimento dos estudantes sobre determinados termos que são importantes para a criação de algoritmos e a programação. O questionário apresentava perguntas sobre: o que é um problema, o que é lógica, o que se entende por algoritmo, termos básicos utilizados na programação e os conhecimentos dos estudantes sobre as linguagens de programação.

Na Questão 4, cuja pergunta foi “Em sua opinião, o que é um problema?”, os estudantes escreveram diversas opiniões, sendo que dois deles demonstraram compreender um pouco sobre o assunto relacionando-o com a programação. O Estudante A escreveu que problema é “*Qualquer coisa que exija uma resposta ou solução*”¹⁰. Percebe-se que o estudante entende como problema algo que necessita

¹⁰ Os comentários dos estudantes são apresentados como citação direta (curta ou longa, conforme forem até 3 linhas ou mais), em letra itálica.

de uma solução, que precisa de uma resposta. Já o Estudante E, apresenta que um problema é algo difícil de ser resolvido, pois não se sabe por onde começar, quando escreve: *“Algo que não foi bem desenvolvido ou algo que não sabemos por onde começar para resolvê-lo”*. Esta é uma das dificuldades para os estudantes que estão iniciando os estudos de algoritmos, não saber como iniciar a resolução de um problema. Os autores Junior, Vieira e Vieira (2015) descreveram em sua pesquisa que, ao citar a capacidade de abstração, que a leitura e a interpretação dos textos dos enunciados, são algumas das dificuldades iniciais quando se está aprendendo a programar.

A Questão 5, “Em sua opinião, o que é lógica?”, teve como intuito conhecer o que os estudantes entendem por lógica, qual o significado de ter lógica na resolução de algum problema. Dois estudantes deixaram essa pergunta em branco, já os outros responderam, com destaque aos mesmos dois estudantes A e E, que demonstraram ter algum conhecimento sobre a lógica envolvida na resolução de problemas. O Estudante A descreve que lógica é *“Algo exato, fixo, que dependa da situação. Em outras palavras, é a forma mais simples de se chegar a um resultado com os dados disponíveis”*. Já o Estudante E cita que lógica é *“Solucionar um problema da melhor forma possível”*. Puga e Riseti (2009, p. 1) descrevem que lógica é a “[...] maneira pela qual um pensamento ou ideia é organizada e apresentada, possibilitando que cheguemos a uma conclusão”. Desta forma, os estudantes escreveram respostas convergentes com os conceitos dos autores anteriormente mencionados.

Com a Questão 6, “O que você entende por algoritmo?”, pretendeu-se saber se os estudantes conheciam o significado da palavra algoritmo. Dois estudantes escreveram *“não sei”*. Dentre os que responderam, destacam-se, além dos Estudantes A e E, que também demonstraram conhecimento sobre este termo, os Estudantes C e G. O Estudante C respondeu que algoritmo é um *“Conjunto de comandos que forma algo”*. Por esta resposta percebe-se que ele tem conhecimento de que algoritmo é um “conjunto de comandos”. Mesmo não tendo explicado o que pode-se fazer com eles, mas que pode-se fazer “algo” por meio dele. Já o Estudante G escreve que algoritmo é *“[...] um mapa de funcionamento de algo”*, querendo dizer que é o processo para resolver algo, como se fosse um passo a passo. Assim como

definido por Forbellone e Eberspächer (2005, p. 3), “um algoritmo pode ser definido com uma sequência de passos que visam a atingir um objetivo bem definido”.

Na Questão 7, “Marque quais dos termos a seguir você já ouviu falar: Tipos de dados, Variáveis, Português estruturado, Portugol, Comandos condicionais, Comandos de repetição, *VisuAlg*”, são apresentados termos usuais quando se trabalha com algoritmos e códigos escritos em português estruturado. Saber se os estudantes já conhecem alguns dos termos utilizados em algoritmos é importante, pois pode-se visualizar se eles têm conhecimento sobre a estrutura básica para a criação de um programa. Como nas questões anteriores, os Estudantes A e E, foram os que apresentaram maior conhecimento sobre os termos apresentados. No entanto, eles não conheciam o “Português estruturado”, tampouco o “*VisuAlg*”, marcando, assim, todos os outros termos apresentados. Dentre os estudantes, somente o Estudante G não marcou nenhuma resposta e isso deve-se ao fato dele não ter desenvolvido o código de nenhum algoritmo e nem conhecer a codificação de alguma linguagem de programação.

Com a Questão 8, “Já ouviu falar em linguagem de programação? Cite o nome das que conhece”, pretendia-se saber quais os nomes das linguagens de programação que os estudantes já ouviram falar ou tinham conhecimento de seu funcionamento. O Estudante A foi o que mais demonstrou conhecer o nome de diferentes linguagens, como citado por ele “*C/C++/C#, Assembly, Basic, Cobol, Fortan, Python, Java, Ruby e Delphi*”. Isso pode estar relacionado ao fato do pai deste estudante ter trabalhado com programação na linguagem C, e o mesmo ter contato com programação desde “*muito novo*”, conforme comentou o estudante durante a primeira aula da prática pedagógica. O Estudante C escreveu “*Sim*”, que já tinha ouvido falar de linguagens de programação, mas não citou o nome de nenhuma. O Estudante F apresentou que não conhece nenhuma linguagem, enquanto os outros estudantes apresentaram que conhecem em torno de três linguagens cada um. As duas mais citadas foram *Java* e *PHP*.

Já na Questão 9, “Escreva algum comando que conheça sobre programação”, somente o Estudante A, escreveu o seguinte trecho de código “`` = *Adiciona um link a uma imagem (HTML)*”. Este código é

relativo a linguagem HTML¹¹, utilizada para criar páginas para a internet, sendo que o código apresentado pelo estudante é funcional, isso é, se for salvo e executado em um navegador de internet ele funcionará. Todos os outros deixaram esta questão em branco. Poder analisar o que os estudantes já conhecem sobre programação é importante para poder planejar as aulas, tendo como base no que eles já sabem, pois se todos já tiverem noções de programação o foco das aulas seria diferente.

Após analisar as respostas apresentadas pelos estudantes, percebeu-se que se trata de uma turma heterogênea na questão de conhecimentos de programação. Para Moraes e Franco (2011, p. 156) uma turma heterogênea é:

[...] compostas por alunos que diferem não só de forma multicultural, mas também em sua capacidade de aquisição de conhecimentos e em suas habilidades. Podendo ser classificada em termos de idade, motivação, inteligência, autodisciplina, conhecimentos das competências, atitudes e interesses.

Além disso, nenhum estudante demonstrou saber programar em alguma linguagem, mas apresentaram conhecimentos gerais sobre lógica e algoritmos. E constatou-se, também, que alguns estudantes não demonstraram inicialmente nenhum conhecimento sobre a criação de algoritmos e dos termos comumente utilizados no desenvolvimento de programas.

Na seção seguinte será feita a análise da prática pedagógica, apresentando os assuntos que foram trabalhados, as percepções deste pesquisador sobre o ensino e aprendizagem dos conceitos e códigos explicados, os comentários e códigos desenvolvidos pelos estudantes.

4.2 Categoria 2 - Contribuições do *Software VisuAlg* no ensino de algoritmos

Esta categoria surgiu a partir da aplicação da prática pedagógica realizada, sendo que com ela pretende-se atingir o objetivo específico “Desenvolver uma proposta pedagógica, com estudantes do Módulo III do curso Técnico em

¹¹ HTML - *HyperText Markup Language*, que significa Linguagem de Marcação de Hipertexto (NETO, 2011).

Informática, que envolva a utilização do *Software VisuAlg* para a construção de algoritmos”, analisando os acontecimentos ocorridos e registrados no diário de campo durante a prática desenvolvida, bem como os códigos e comentários realizados pelos estudantes.

Ao iniciar a prática pedagógica foram apresentadas algumas situações nas quais poderiam ser utilizados algoritmos para resolvê-las, para situar os estudantes aos assuntos que seriam trabalhados no decorrer das aulas. Para continuar com a prática, apresentou-se aos estudantes os tipos primitivos de dados que são utilizados para a criação dos algoritmos. Os principais tipos de dados trabalhados foram: INTEIRO, REAL, CHARACTER e LÓGICO¹² (PUGA; RISSETTI, 2016, p. 20). Ao serem feitas as explicações destes assuntos, foram apresentados exemplos e solicitado aos estudantes que contribuíssem para complementar a explicação. O Estudante D comentou que para o cálculo de uma média seria necessário utilizar o tipo REAL, pois este aceitaria números com vírgula na resposta. Isso demonstra que este estudante está fazendo a relação entre o que foi trabalhado e o que ele já sabia.

No ensino dos algoritmos costuma-se começar pela escrita da “descritiva narrativa” da resolução do problema. Conforme destacado na seção 2.2, este tipo de algoritmo mais genérico, apresentada pelos autores Guedes (2014) e Puga e Rissetti (2016), consiste em “analisar o problema e escrever, utilizando uma linguagem natural, os processos para a sua resolução”. Para os estudantes conhecerem o processo de criação de algoritmos foi utilizada a descrição narrativa, para resolver os exercícios apresentados na Figura 08 (APÊNDICE J, Enunciado 3).

¹² Para dar maior destaque aos comandos e códigos dos algoritmos, utilizados na escrita do texto, estes foram apresentados em letra maiúscula, para diferenciar os comandos das variáveis. Mas conforme apresentado por Souza (2009) todos os comandos e códigos podem ser escritos em minúsculo ou maiúsculo no *VisuAlg*.

Figura 08 - Exercícios sobre descrição narrativa.

3) Desenvolva os algoritmos para as seguintes tarefas:

- A. Tomar um banho.
- B. Você é porteiro de uma festa e somente pessoas autorizadas podem entrar.
Selecionar quem pode entrar.
- C. Calcular o volume de uma sala.
- D. Limpar e organizar o quarto.
- E. Ajustar a temperatura ideal da água do banho.

Fonte: O AUTOR (2016).

Nestes exercícios os estudantes tiveram que descrever procedimentos para a resolução dos problemas apresentados. Para facilitar o entendimento, o Exercício A (*Tomar um banho*) foi resolvido com a turma como exemplo. A seguir apresenta-se uma possível solução para o problema, que foi apresentada pelo Estudante A:

- Abrir a porta.
- Ligar a luz.
- Entrar.
- Fechar porta.
- Conferir objetos.
- Tirar roupa.
- Entrar no box.
- Ligar chuveiro.
- Tomar banho.
- Desligar chuveiro.
- Secar.
- Colocar roupa.
- Sair do banheiro.

Esta atividade de criação de algoritmos, na forma de descrição narrativa, todos os estudantes conseguiram realizar, pois este método não se preocupa tanto com a codificação, mas sim com o entendimento do que foi escrito. O importante nesta construção é que qualquer pessoa que leia este tipo de algoritmo saiba seguir o seu processo para resolução do problema proposto. Guedes (2014) apresenta que a principal vantagem de utilizar a descrição narrativa é de não precisar aprender nenhum conceito novo, e a desvantagem é que a escrita na “língua natural” pode gerar várias interpretações, sendo algumas vezes difícil a sua adaptação posterior para uma linguagem de programação.

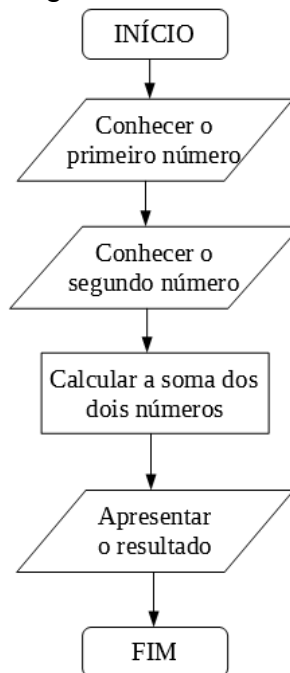
Outro processo importante para o ensino da lógica de programação é a criação dos algoritmos por meio de fluxograma, que consiste em representar a resolução de um problema com “símbolos gráficos predefinidos” (GUEDES, 2014, p. 7). Inicialmente foi descrito o problema “a) Faça um algoritmo que leia dois números

inteiros e calcule a soma” (APÊNDICE K, Enunciado 3), como exemplo. Este Exercício A foi resolvido em aula utilizando-se a descrição narrativa que os estudantes já conheciam. Uma solução para este problema foi construída em conjunto com a turma, como observa-se a seguir:

- Conhecer o primeiro número.
- Conhecer o segundo número.
- Calcular a soma dos dois números.
- Apresentar o resultado (O AUTOR, 2016).

Logo após este algoritmo foi representado por meio de fluxograma, com destaque aos símbolos e setas que representam os fluxos do código, comparando-os com a descrição narrativa escrita anteriormente. Os estudantes utilizaram o editor de texto *LibreOffice Writer* para desenhar os seus fluxogramas, conforme observa-se na representação feita pelo Estudante C, apresentada na Figura 09.

Figura 09 - Representação gráfica do Exercício A feita em fluxograma.



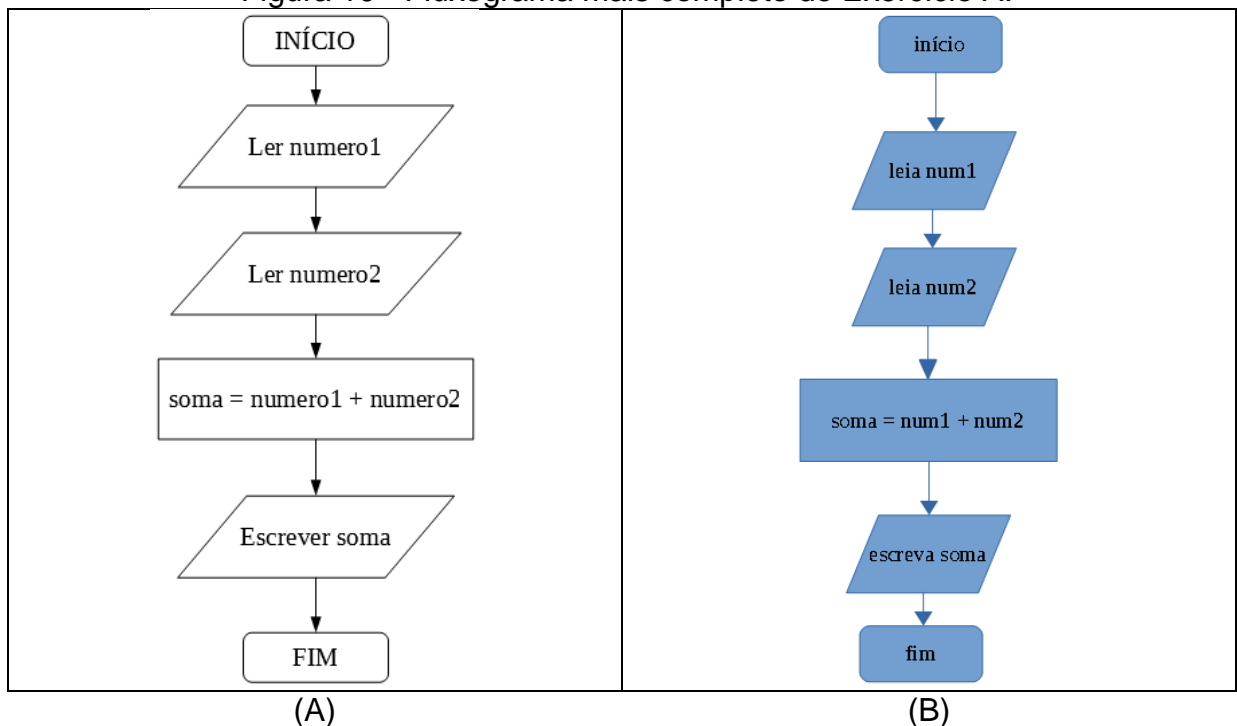
Fonte: Estudante C (2016), a partir do *Software LibreOffice Writer*.

Como pode-se perceber, no fluxograma da Figura 09, são apresentados os mesmos quatro passos que foram escritos na descrição narrativa, mas agora com identificadores de INÍCIO e FIM do algoritmo e alguns símbolos para representarem a interação com o usuário (paralelogramo) e processamento do sistema (retângulo). Estes símbolos foram adaptados dos apresentados por Guedes (2014, p. 7-8) e Puga e Rissetti (2016, p. 14-15), que descrevem que este tipo de representação é

considerado “um algoritmo universal” e está documentado na norma ISO 5807:1985, como padrão internacional.

Na Figura 10A e 10B foram realizadas algumas alterações, utilizando-se agora os conceitos de variáveis para guardar as informações dos números e a soma.

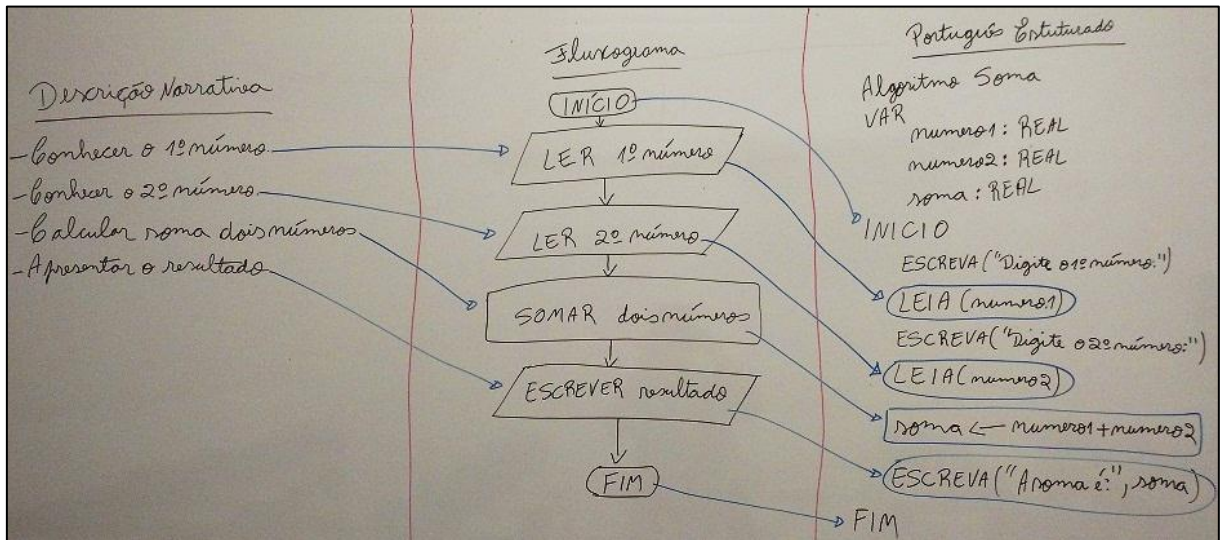
Figura 10 - Fluxograma mais completo do Exercício A.



Fonte: (A) Estudante C (2016) e (B) Estudante E (2016), a partir do *Software LibreOffice Writer*.

Para iniciar a codificação do algoritmo por meio do Português Estruturado, teve-se como base o fluxograma apresentado na Figura 10A, comparando-se os símbolos e fluxos com a codificação utilizada pelo *Software VisuAlg*. As três representações para o mesmo algoritmo, foram escritas no quadro uma ao lado da outra (FIGURA 11), comparando-se cada etapa criada com a anterior.

Figura 11 - Comparação entre Descrição Narrativa, Fluxograma e Português Estruturado.



Fonte: O Autor (2016).

Esta comparação é importante para que os estudantes possam compreender que, pode-se iniciar a criação dos algoritmos, a partir da descrição narrativa e só após criar o fluxograma e posterior codificação deste para Português Estruturado ou para alguma linguagem de programação.

Guedes (2014) acredita que, com estes comparativos, os estudantes podem compreender melhor a relação entre a leitura de um problema computacional e a sua possível resolução por meio de “pseudocódigo”, o português estruturado. Para isso, inicialmente o autor apresenta vários exemplos de algoritmos representados por meio da descrição narrativa, fluxograma e português estruturado. Acredita-se que dessa forma os estudantes possam assimilar e associar, escrevendo ou mentalmente, a relação entre a construção do algoritmo de forma descritiva com o código necessário para a sua resolução, ao criar uma sequência lógica para resolver às diferentes estruturas de representação.

Ao iniciar a resolução dos exercícios propostos na Figura 12 (APÊNDICE K, Enunciado 3) foi apresentado o problema “c) Faça um algoritmo que solicite o nome do usuário e a sua idade, no final escreva o nome e idade da pessoa”, para ser resolvido com o auxílio do *VisuAlg*.

Figura 12 - Exercícios sobre fluxograma.

3) Desenvolva os algoritmos solicitados abaixo utilizando o Software VisuAlg:

- a) Faça um algoritmo que leia dois números inteiros e calcule a soma.
- b) Faça um algoritmo que leia dois números inteiros, nas variáveis X e Y. Troque os números entre as variáveis e escreva na tela.
- c) Faça um algoritmo que solicite o nome do usuário e a sua idade, no final escreva o nome e a idade da pessoa.
- d) Faça um algoritmo que receba 3 notas de um estudante e escreva a média no final.
- e) Faça um algoritmo que calcule o volume de uma sala e escreva o resultado.
- f) Faça um algoritmo que calcule as quatro operações (+, -, * e /) entre dois números e escreva o resultado.
- g) Faça um algoritmo que solicite a área de uma parede (altura e largura), solicite a área de uma das faces de um tijolo e escreva quantos tijolos são necessários para construir a parede.

Fonte: O AUTOR (2016).

Os estudantes tiveram poucas dúvidas sobre a resolução deste problema. As duas principais foram em relação ao melhor tipo de dado para armazenar a idade, sendo explicado que é o INTEIRO, pois normalmente não são armazenadas idades “quebradas”. A outra dúvida foi sobre como escrever o nome das variáveis compostas por duas palavras, sendo que é recomendado que fossem escritas juntas (por exemplo: *nomedousuario*), sem espaço em branco entre elas, pois o *software* pode reconhecer como duas variáveis diferentes se forem escritas com espaço entre elas.

Durante o desenvolvimento deste algoritmo percebeu-se que um estudante construiu o ESCRIVA de forma bem completa, utilizando palavras para completar o resultado escrito na tela, conforme pode ser visto no trecho de código escrito pelo Estudante B:

```
escreva (nome, " você tem ", idade, " anos de idade.")
```

Dentre as principais dúvidas encontradas pelos estudantes durante a resolução dos exercícios apresentados na Figura 12, pode-se destacar:

- O Estudante G perguntou por que não estava conseguindo visualizar o resultado de um cálculo e constatou-se que ele esqueceu um ESCRIVA para mostrar o resultado. O seu código estava correto, por isso não eram apresentados erros.

- O Estudante D estava com problema no resultado do cálculo da média, que apresentava erro. Após analisar o código criado descobriu-se que ele utilizava a barra que representa a operação de divisão invertida, com isso o cálculo não era feito corretamente.
- O Estudante F fez uma pergunta interessante, que foi apresentada durante a correção dos exercícios. Ele perguntou: “*Posso fazer a soma de um número inteiro com outro real?*”. Com esta pergunta percebe-se que o estudante compreende a ideia de que os tipos de dados primitivos, utilizados para números, têm formatos diferentes. Foi explicado que pode ser realizada a soma entre tipos de dados distintos, mas para salvar o resultado numa variável, esta deverá aceitar os tipos de dados das duas anteriores. Por exemplo, para somar uma variável INTEIRO com outra REAL, a variável resultante deverá ser do tipo REAL.

O exercício que gerou mais dúvidas foi o apresentado na Questão G (APÊNDICE K, Exercício 3), “g) Faça um algoritmo que solicite a área de uma parede (altura e largura), solicite a área de uma das faces de um tijolo e escreva quantos tijolos são necessários para construir a parede”, gerando bastante discussão sobre qual a melhor forma para resolvê-lo. Não pela complexidade do código a ser escrito, mas sim em como fazer os cálculos, quais as fórmulas eram necessárias. Alguns estudantes digitaram as medidas da parede e do tijolo em centímetros, outros em metros e fizeram a conversão, já o Estudante E solicitou a parede em metros e o tijolo em centímetros, fazendo assim as conversões conforme pode-se observar na Figura 13.

Figura 13 - Resposta do Estudante E para a Questão G.

```

algoritmo "Tijolos"

var
    altParede : real
    compParede : real
    altTijolo : real
    compTijolo : real
    areaParede : real
    areaTijolo : real
    tijolos : real

inicio
    escreva("Altura parede (m):")
    leia(altParede)
    escreva("Comprimento parede (m):")
    leia(compParede)
    escreva("Altura tijolo (cm):")
    leia(altTijolo)
    escreva("Comprimento tijolo (cm):")
    leia(compTijolo)

    areaParede <- altParede * compParede
    areaTijolo <- (altTijolo/100) * (compTijolo/100)
    tijolos <- (areaParede/areaTijolo)

    escreva("Serão necessários: ",tijolos," tijolos.")

finalgoritmo

```

Fonte: Estudante E (2016), a partir do *Software VisuAlg*.

Como pode-se perceber, na Figura 13, o Estudante E solicitou a digitação as medidas da parede em metros e as do tijolo em centímetros. Com isso, realizou o cálculo da área da parede, converteu as medidas do tijolo para metros, calculou a área do tijolo e após isso determinou quantos tijolos eram necessários para construir a parede.

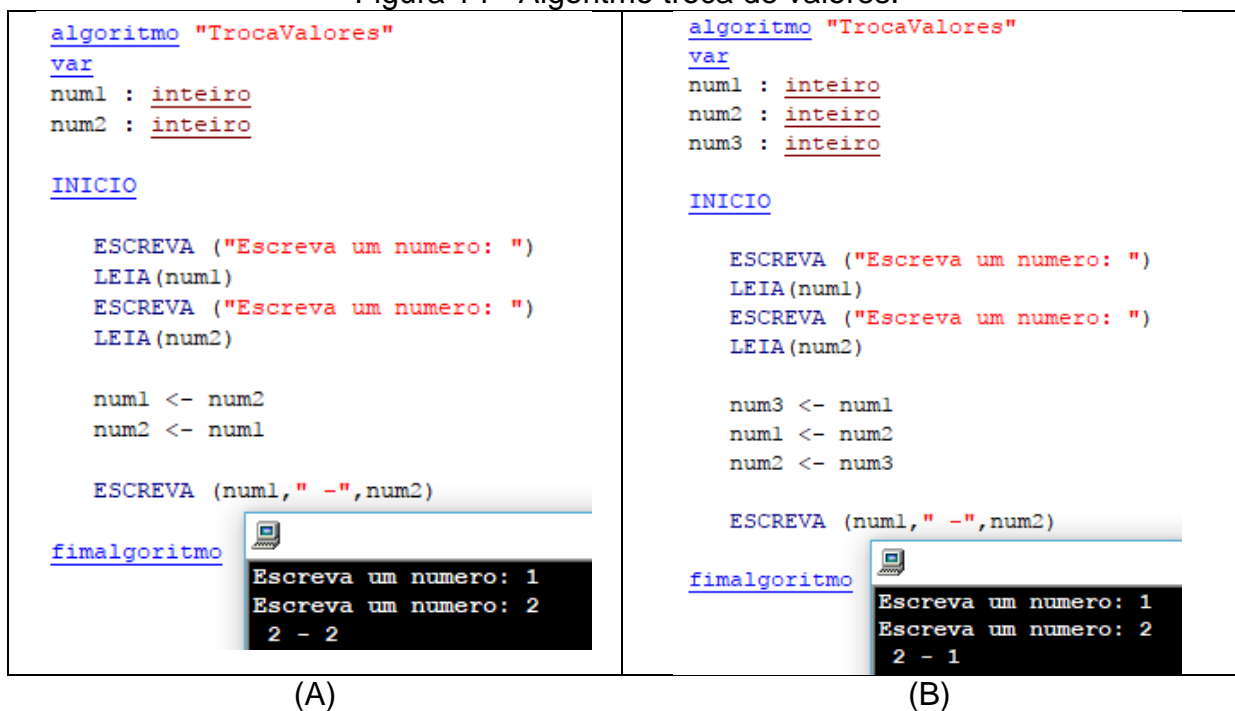
Com a observação de discussões como esta pode-se perceber que em alguns casos a dificuldade dos estudantes está em saber qual função utilizar para realizar o cálculo, não na construção do código para o algoritmo. Esta afirmação vai ao encontro do que Moreira e Ynogutti (2012, p. 105) dizem, quando apresentam que:

Outro fator que pode influenciar na solução de um problema é a falta de raciocínio lógico necessário para a solução. O aluno pode simplesmente compreender o que precisa ser feito, porém não consegue conceber a solução. Um dos fatores deste problema pode estar ligado à falta de base do aluno, visto que a disciplina de algoritmos está ligada à base matemática que vem do ensino médio.

Ao analisar o processo de resolução dos exercícios apresentados na Figura 12 utilizando o *VisuAlg*, pode-se perceber que os estudantes conseguiram desenvolver o código do algoritmo de forma satisfatória com poucos erros e dúvidas. Somente o Estudante G apresentou maior dificuldade, na construção de comandos básicos como o **ESCREVA** e cálculo da média. Assim, foi necessária uma explicação mais detalha, mostrando passo a passo o funcionamento e sintaxe dos comandos vistos até o momento.

Para exercitar um pouco mais os comandos **LEIA** e **ESCREVA**, e principalmente a utilização das variáveis, foi proposta a resolução do exercício “Faça um algoritmo que leia dois números e inverta os valores das variáveis”, que não estava dentre os apresentados anteriormente. Entender a resolução deste problema é importante, pois os estudantes precisam utilizar uma variável auxiliar para não “perder”¹³ um dos valores, sendo que isso não está descrito no problema. A Figura 14A mostra o algoritmo criado pelo Estudante C, sendo que na resposta (em destaque) pode-se observar que o número 1 digitado anteriormente não aparece.

Figura 14 - Algoritmo troca de valores.



(A)

(B)

Fonte: Estudante C (2016), a partir do *Software VisuAlg*.

¹³ Em programação, perder o valor de uma variável pode acontecer quando é atribuído um novo valor a ela, sobrescrevendo o valor que já estava armazenado. Com isso, perdendo o valor armazenado anteriormente.

A turma foi orientada que, para trocar os valores entre as variáveis, é necessário guardá-lo em uma variável auxiliar para ele não seja substituído pelo valor da outra. A Figura 14B mostra o algoritmo do mesmo Estudante C após a correção feita.

Um código considerado importante para a criação de algoritmos é o condicional SE (simples, composto e encadeado), que é utilizado para testar se uma condição é verdadeira ou falsa. Puga e Rissetti (2016, p. 43) descrevem que:

A estrutura de seleção composta prevê dois conjuntos de instruções para serem realizadas de acordo com a avaliação da condição: um conjunto de instruções que será executado quando a condição obtiver o resultado verdadeiro e um conjunto de instruções para o resultado falso.

Para melhor compreensão dos estudantes sobre este assunto, apresentou-se o problema: “Faça um algoritmo que leia duas notas, calcule a média e escreva se a nota equivale a Aprovado ou Reprovado”. Para esta situação é necessário utilizar o comando SE com o SENAO, sendo o teste do SE relativo ao Aprovado e do SENAO ao Reprovado.

A única dúvida que surgiu na resolução deste problema é se, para uma média de aprovação 6,0, o teste do Aprovado poderia ser “*media > 5*”. Pelo entendimento de alguns estudantes era só testar se a média era maior do que 5, não considerando que poderiam existir notas decimais com vírgula. Com isso, existindo várias opções de notas entre 5,0 e 6,0 que também são consideradas como reprovadas. A explicação consistiu de que o cálculo da média pode gerar números reais (com vírgula), então a nota poderia ficar 5,9 sendo Reprovado. Com isso, concluiu-se em conjunto com os estudantes que o teste para o Aprovado teria que ser “*media >= 6,0*”.

Para exercitar os conhecimentos sobre o comando SE, foi proposto a resolução dos exercícios da Figura 15 (APÊNDICE L).

Figura 15 - Exercícios sobre SE utilizando o *Software VisuAlg*.

Exercite suas habilidades desenvolvendo os algoritmos solicitados a seguir:

- 1) Faça um algoritmo que leia uma letra e diga se esta equivale ao sexo feminino ou masculino.
- 2) Faça um algoritmo que leia dois números e escreva qual deles é o maior.
- 3) Faça um algoritmo que leia três números e escreva qual deles é o maior.
- 4) Faça um algoritmo que simule uma calculadora simples.
- 5) Faça um algoritmo que receba um número inteiro e verifique se este número é par ou ímpar.
- 6) Faça um algoritmo que leia um número e escreva se ele é: positivo, negativo ou zero.
- 7) Faça um algoritmo que leia o dia, o mês e o ano separadamente, apresente se esta data é válida ou não e escreva como no exemplo.
Exemplo: 25/08/2016.
- 8) Faça um algoritmo que leia dois número (X e Y), troque os valores entre as variáveis e escreva na tela os valores trocados.
- 9) Faça um algoritmo que leia um ano e determine se ele é bissexto.
- 10) Elabore um algoritmo que efetue o cálculo do reajuste de salário de um funcionário. Considere que o funcionário deverá receber um reajuste:
 - salário < 500 - 15%
 - salário >= 500 e <= 1000 - 10%
 - salário > 1000 - 5%
- 11) Faça um algoritmo que leia dois números. Se todos forem pares efetue a soma deles, caso todos sejam ímpares efetue a multiplicação e em caso contrário, subtraia um pelo outro. O algoritmo deve ainda emitir mensagens do tipo:
 - Todos pares, a soma é:
 - Todos ímpares, a multiplicação é:
 - Ambos os tipos, a subtração é:
- 12) Faça um algoritmo que leia a sigla de um estado da região sul e escreva o nome deste estado por extenso.
- 13) Leia um número real. Se o número for positivo imprima a raiz quadrada. Do contrário, imprima o número ao quadrado.

Fonte: O AUTOR (2016).

A Questão 1 (Faça um algoritmo que leia uma letra e diga se esta equivale ao sexo feminino ou masculino) o Estudante D já quis fazer testes com três opções. Ele queria apresentar as respostas “Feminino”, “Masculino” ou “Letra inválida”, mas o seu código estava apresentando erro, como pode ser visualizado na Figura 16.

Figura 16 - Algoritmo do Estudante D com erro no teste de três opções de resposta.

```

algoritmo "TestaLetra"

var
    sexo : caracter
inicio

    escreva("Digite a letra correspondente ao seu sexo (M/F): ")
    leia(sexo)

    se (sexo = "M") entao
        escreva("Masculino")
    senao (sexo = "F")
        escreva("Feminino")
    senao
        escreva("Letra inválida")
    fimse

fimalgoritmo

```

Fonte: Estudante D (2016), a partir do *Software VisuAlg*.

Assim, para resolver este problema, orientou-se que a cada duas opções de respostas é necessário um teste com SE e um SENAO, já para três opções são necessários dois SE e dois SENAO. Após a explicação, o Estudante D fez as alterações necessárias para corrigir o código e chegou ao algoritmo que pode ser visto na Figura 17.

Figura 17 - Algoritmo do Estudante D após a correção.

```

algoritmo "TestaLetra"

var
    sexo : caracter
inicio

    escreva("Digite a letra correspondente ao seu sexo (M/F): ")
    leia(sexo)

    se (sexo = "M") entao
        escreva("Masculino")
    senao
        se (sexo = "F")
            escreva("Feminino")
        senao
            escreva("Letra inválida!")
        fimse
    fimse

fimalgoritmo

```

Fonte: Estudante D (2016), a partir do *Software VisuAlg*.

Observa-se na Figura 17 que o Estudante D compreendeu a explicação e conseguiu fazer o teste para apresentar as três opções de respostas que ele desejava.

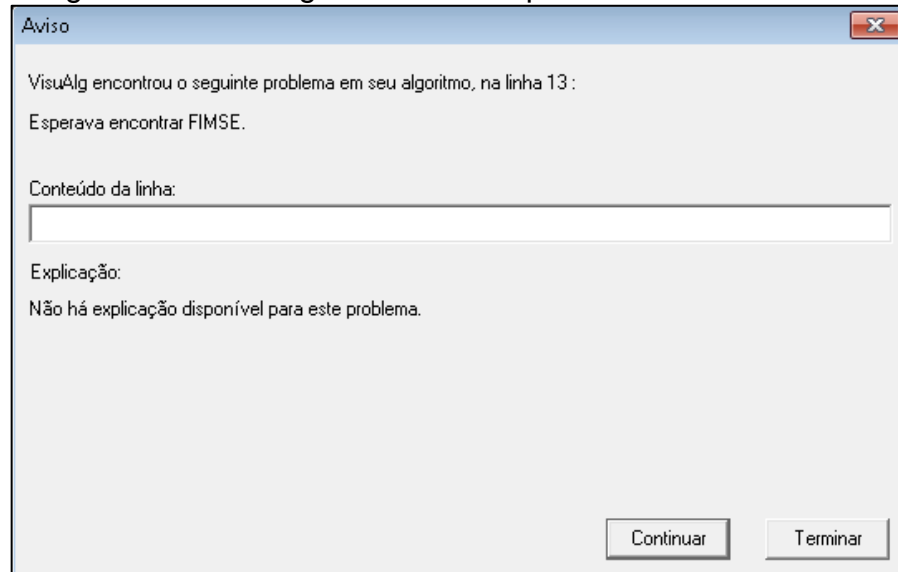
Muitas vezes colocar comandos concatenados, como o apresentado na Figura 17, pode confundir os estudantes, pois acaba ficando um pouco confuso a estrutura do código. Para isso, uma função interessante disponível no *VisuAlg* é a indentação¹⁴ das linhas dos comandos, deixando o algoritmo legível e mais fácil para identificar visualmente qual código está dentro de outro. A opção de indentação auxilia, não somente a quem está aprendendo a programar, a se organizar e visualizar melhor o código, mas também a programadores a desenvolver códigos de maneira que todos possam compreender posteriormente.

Vale destacar que os estudantes não tiveram muitas dúvidas em como resolver os exercícios do Apêndice L. As principais perguntas eram como fazer os testes e qual a equação usar, como perguntado pelo Estudante D - “*As fórmulas o professor passa?*”.

Além das dúvidas já apresentadas, outra situação de destaque é que alguns estudantes esquecem-se de fechar os comandos, como por exemplo, o SE que precisa do FIMSE ao final para funcionar corretamente. Um aspecto favorável da utilização do *VisuAlg* é que ele apresenta mensagens informando que esperava encontrar o comando de fechar, como pode ser visualizado na Figura 18.

¹⁴ A indentação organiza o código-fonte e coloca “[...] os comandos dentro de uma estrutura de 3 colunas à direita da coluna inicial da estrutura conforme a configuração padrão” (TONET, 2017. p. 6).

Figura 18 - Mensagem de erro: Esperava encontrar FIMSE.



Fonte: O Autor (2016), a partir do *Software VisuAlg*.

Por meio de mensagens, como a visualizada na Figura 18, os estudantes conseguem acompanhar se o código digitado está correto e, também, encontrar dicas de como resolver o erro. Com isso, pode-se ter uma referência na hora de desenvolver os algoritmos propostos, por meio das mensagens que o *Software VisuAlg* apresenta.

À medida que os estudantes terminavam de resolver os exercícios propostos, eles iam se ajudando na resolução dos problemas e nas dúvidas encontradas, tentavam também fazer alterações no código para ver qual o resultado que apareceria. Uma situação interessante que aconteceu foi quando dois estudantes estavam conversando e rindo, isso chamou a atenção dos outros para verem o que era. Ao conferir o que eles estavam fazendo, percebeu-se que haviam criado um novo algoritmo, além daqueles propostos, que solicitava que fossem digitadas algumas informações e mostrava um resultado. O algoritmo criado foi nomeado, pelos estudantes, como “*Testando*”, conforme pode ser visto na Figura 19.

Figura 19 - Algoritmo “Testando”, criado pelos estudantes.

```

algoritmo "Testando"

var
    nome : caracter
    idade : inteiro
    trabalho : caracter

inicio

    escreval ("BEM-VINDO AO TESTANDO...")

    escreva ("Qual o seu nome? ")
    leia (nome)

    escreval (nome, ", este nome parece meio estranho.")

    escreva ("Qual a sua idade? ")
    leia (idade)

    se (idade<18) entao
        escreval ("Idade de bebê?")
    senao
        escreval ("Idade OK!")
    fimse

    escreva ("Você trabalha (S/N)? ")
    leia (trabalho)

    se (trabalho="N") entao
        escreval ("Folgado!")
    senao
        escreval ("Tudo certo!")
    fimse

finalgoritmo

```

Fonte: Estudantes B e F (2016), a partir do *Software VisuAlg*.

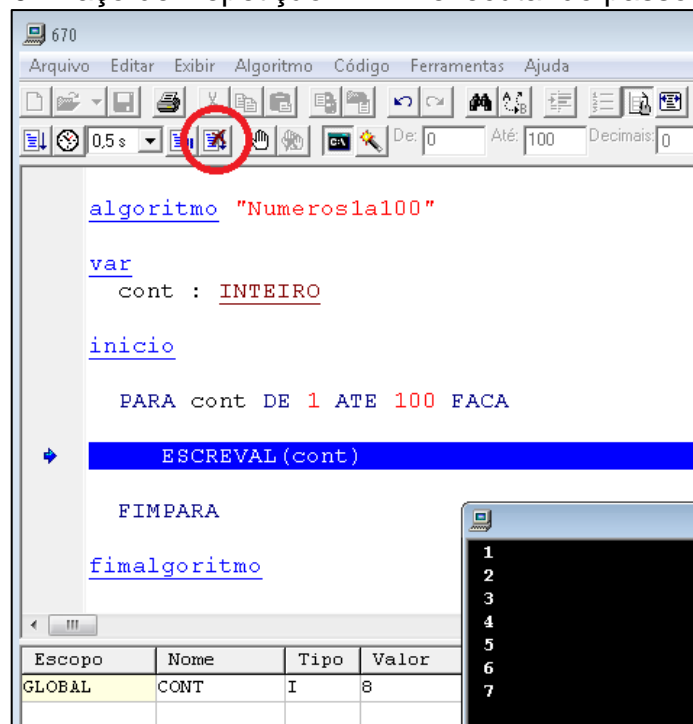
É por meio de momentos como este que os estudantes criam significado para aquilo que estão aprendendo, pois compreenderam o que foi explicado e propõem diferentes formas para representação do conhecimento adquirido. A concepção de Moran, Masetto e Behrens (2007) é um dos aportes teóricos para compreender o processo de aprendizagem dos estudantes:

Aprendemos quando descobrimos novas dimensões de significação que antes se nos escapavam, quando vamos ampliando o círculo de compreensão do que nos rodeia [...]. Aprendemos mais quando estabelecemos pontes entre a reflexão e a ação, entre a experiência e a conceituação, entre a teoria e a prática; quando ambas se alimentam mutuamente (MORAN; MASETTO; BEHRENS, 2007, p. 23).

Outros dois comandos importantes para a criação de algoritmos, mais complexos, são os Laços de Repetição (PARA e ENQUANTO) que servem para

criar *loopings*¹⁵ e repetir diferentes comandos ou mostrar mensagens. Para testar o correto funcionamento dos Laços de Repetição, o *VisuAlg* dispõe de uma ferramenta que executa o código passo a passo (linha por linha), continuando somente após ser pressionado o botão correspondente. A Figura 20 mostra um algoritmo sendo executado com a função de passo a passo, com destaque em vermelho para o botão que deve ser pressionado.

Figura 20 - Laço de Repetição PARA executando passo a passo.



Fonte: O Autor (2016), a partir do Software *VisuAlg*.

Esta opção é muito útil para que os estudantes possam acompanhar a execução de seu código, e conferir qual o local que está acontecendo algum erro, ou quais os valores que as variáveis estão recebendo. Com isso, pode-se ter um maior controle do que é feito em cada linha do algoritmo.

Nos exercícios sobre Laços de Repetição (APÊNDICE M e N), percebeu-se que os estudantes costumam esquecer que com o PARA o próprio comando controla o incremento no contador, já com o ENQUANTO é necessário que sejam inseridas duas linhas de código a mais para controlar o número de início do contador

¹⁵ “Os laços de repetição também são conhecidos por sua tradução em inglês: *loops* ou *looping*. Ganham esse nome por lembrarem uma execução finita em círculos, que depois segue seu curso normal” (FORBELLONE; EBERSPÄCHER, 2005, p. 48).

e outra para o incremento, conforme pode ser comparado na Figura 21A e 21B, respectivamente.

Figura 21 - Comparação entre o PARA e o ENQUANTO.

<pre> PARA cont DE 1 ATE 5 FACA FIMPARA </pre>	<pre> cont <- 1 ENQUANTO (cont <= 5) FACA cont <- cont + 1 FIMENQUANTO </pre>
---	--

(A)

(B)

Fonte: O Autor (2016), a partir do *Software VisuAlg*.

Caso o estudante esqueça-se de colocar o contador “`cont <- cont + 1`” (Figura 21B), o laço de repetição ENQUANTO cria um *looping* infinito. Ou seja, o algoritmo não iria mais parar sua execução, sendo necessário interromper manualmente o fluxo.

Para saber se os estudantes estavam compreendendo a criação dos algoritmos propostos, foi solicitado que eles analisassem o código apresentado na Figura 22 e identificassem o erro contido nele.

Figura 22 - Exercício de Laço de Repetição PARA com erro.

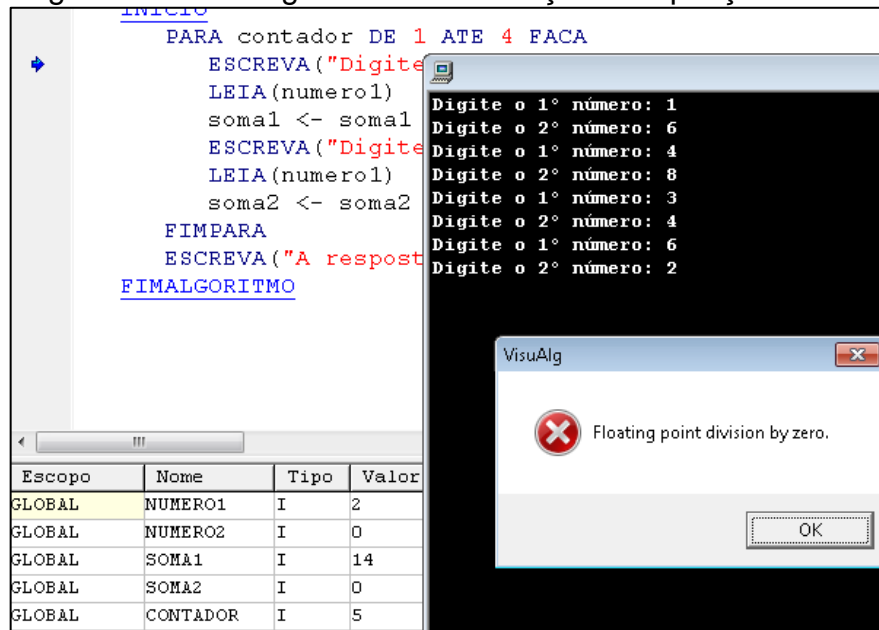
```

ALGORITMO "Exercicio06"
VAR
    numero1 : INTEIRO
    numero2 : INTEIRO
    soma1 : INTEIRO
    soma2 : INTEIRO
    contador : INTEIRO
INICIO
    PARA contador DE 1 ATE 4 FACA
        ESCRIVA("Digite o 1º número: ")
        LEIA(numero1)
        soma1 <- soma1 + numero1
        ESCRIVA("Digite o 2º número: ")
        LEIA(numero1)
        soma2 <- soma2 + numero2
    FIMPARA
    ESCRIVA("A resposta é: ", soma1 / soma2)
FIMALGORITMO
    
```

Fonte: O Autor (2016), a partir do *Software VisuAlg*.

O algoritmo apresentado na Figura 22 está funcional, o único problema é que ele apresenta a mensagem informando que estava sendo feita uma divisão por zero, conforme pode ser visualizado na mensagem em inglês da Figura 23.

Figura 23 - Mensagem de erro do Laço de Repetição PARA.



Fonte: O Autor (2016), a partir do Software VisuAlg.

Primeiramente os estudantes executaram o código para ver qual o erro era apresentado, logo após conferiram o algoritmo para encontrar o problema. Os Estudantes A, B e E encontraram rapidamente o problema/erro do código, pois conferiam o conteúdo das variáveis apresentadas pelo *VisuAlg*, e perceberam que as variáveis “numero2” e “soma2” estavam com zero de valor (Figura 23). Com isso, notaram que o segundo LEIA estava lendo o “numero1” novamente e que o correto seria ler o “numero2”. Já os outros estudantes demoraram um pouco mais para encontrar o problema.

Outro algoritmo que os estudantes analisaram está representado na Figura 24, sendo que neste os estudantes precisavam informar o que o código faz e descobrir o erro contido neste.

Figura 24 - Exercício de Laço de Repetição ENQUANTO com erro.

```

ALGORITMO "Exercicio05"
VAR
    numero : INTEIRO
    contador : INTEIRO
    soma : INTEIRO
    resultado : INTEIRO
INICIO
    contador <- 0
    resultado <- 0

    ENQUANTO (numero >= 0) FACA
        ESCREVA("Informe um número: ")
        LEIA(numero)
        SE (numero >= 0) ENTAO
            contador <- contador + 1
            soma <- soma + numero
        FIMSE
    FIMENQUANTO

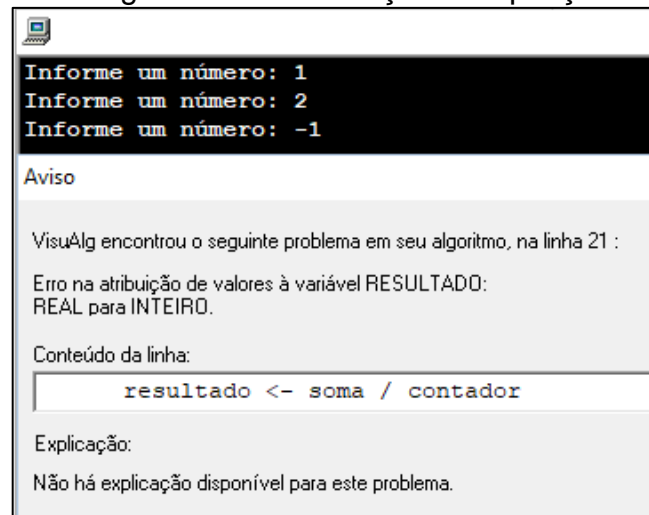
    resultado <- soma / contador
    ESCREVA("A média é: ", resultado)
FIMALGORITMO

```

Fonte: O Autor (2016), a partir do *Software VisuAlg*.

Os estudantes testaram o código para ver o seu funcionamento e inicialmente não apareceu nenhum erro, só que o algoritmo ficava rodando e solicitando números sem parar. Foi quando o Estudante D, ao analisar melhor o código, percebeu pelo teste "ENQUANTO (numero >= 0) FACA" que para encerrar a execução era necessário digitar um número negativo. Ao continuar os testes perceberam que assim que o laço de repetição era encerrado, apresentava erro no código, conforme apresentado na Figura 25.

Figura 25 - Mensagem de erro do Laço de Repetição ENQUANTO.



Fonte: O Autor (2016), a partir do *Software VisuAlg*.

Ao analisar a mensagem de erro do *VisuAlg* percebeu-se que são apresentadas dicas de como corrigi-lo. Com isso, facilitou aos estudantes encontrarem o erro e corrigir o problema. Os Estudantes B e D comentaram que como é uma divisão, o resultado poderia conter números com vírgula, sendo necessário que a variável "resultado" seja do tipo REAL.

Exercícios como os apresentados nas Figuras 21 e 23 são importantes para que os estudantes possam demonstrar que estão entendendo a criação e saibam fazer a correção nos códigos que apresentarem problema. Os autores Hostins e Raabe (2007, p. 98) defendem que as ferramentas utilizadas para o ensino de algoritmos precisam permitir

[...] um maior grau de autonomia ao aluno fornecendo recursos de interface que reduzem os enganos iniciais e que possibilitassem acompanhar detalhadamente a execução do programa, além de incluir um mecanismo simples de verificação da correção da solução.

Percebeu-se que a cada exercício que os estudantes iam desenvolvendo, menos auxílio eles precisavam. Isso deve-se a eles ajudarem uns aos outros a esclarecer as suas dúvidas e, também, por estarem compreendendo os comandos que foram estudados para a criação dos códigos dos algoritmos. Leite et al. (2013, p. 640) também apresentam seus resultados: “[...] pode-se observar que os estudantes de Programação [com o uso do *VisuAlg*] puderam aprender de maneira mais prática, rápida e agradável os conceitos de algoritmos”. Esse mesmo aprendizado percebeu-se nos estudantes participantes da prática pedagógica desenvolvida.

Além disso, como já descrito na seção 2.2, diferentes autores como Souza (2009), Costa (2010), Almeida (2013), Leite et al. (2013), Guedes (2014) e Puga e Rissetti (2016) defendem o uso de alguma ferramenta para auxiliar no ensino de algoritmos, e citam o *VisuAlg* como um bom recurso para quem está iniciando na criação de algoritmos ou na programação. Costa (2010) ainda descreve que os recursos didáticos disponíveis auxiliam no ensino dos comandos em português, pois foram pensados justamente para auxiliar no processo de ensino da lógica de programação.

4.3 Categoria 3 - Percepções dos estudantes sobre o *Software VisuAlg*

A presente categoria surgiu das respostas concedidas pelos estudantes por meio do Questionário - Avaliação da Prática Pedagógica (APÊNDICE O), realizado ao final da prática pedagógica. O questionário teve por objetivo analisar a impressão, os comentários e o aprendizado dos estudantes após construírem diferentes tipos de algoritmos com o auxílio do *Software VisuAlg*. Esta categoria vem ao encontro do objetivo específico “Analisar como a utilização do *Software VisuAlg* contribui para a aprendizagem da lógica de programação”, proposto ao iniciar a pesquisa sobre este assunto. Vale ressaltar que o Estudante G, aparentemente o que apresentava maior dificuldade no desenvolvimento dos algoritmos, não estava presente nesta aula.

Na Questão 01 - “Qual forma de construção dos algoritmos você preferiu, digitando no editor de texto ou utilizando o *Software VisuAlg*? Quais as diferenças que você percebeu ao escrever os algoritmos das duas maneiras?”, todos os estudantes escreveram que preferem utilizar o *VisuAlg*. Percebeu-se que os estudantes tiveram uma boa impressão com a utilização deste *software*, pois descreveram a facilidade de criar os algoritmos, poder testá-los e ver os erros apresentados, conforme descrito pelo Estudante E: “*A forma que eu preferi foi a utilização do Software VisuAlg, pois com ele caso ocorra algum erro o software irá informar e já no editor de texto não se tem certeza absoluta se o algoritmo está totalmente correto*”. O mesmo é percebido na resposta do Estudante C, quando ele descreve que achou mais fácil aprender ao utilizar o *VisuAlg*: “*Preferi pelo Software VisuAlg, pois foi uma maneira mais fácil de aprender*”.

Souza (2009) corrobora com as impressões apresentadas pelos estudantes com o uso do *VisuAlg* quando descreve que:

A experiência mostra que ele facilita o entendimento de como um programa de computador funciona, possibilita um *feedback* imediato sobre a correção e exatidão do pseudocódigo digitado, e o que se considera mais importante, instiga o aluno a experimentar e ver o resultado de suas alterações imediatamente (SOUZA, 2009, p. 8).

Com a Questão 02 - “Aprender os algoritmos na Língua Portuguesa auxiliou ou não na compreensão dos códigos e comandos? Explique por que.”, pretendeu-se

saber qual a percepção dos estudantes em desenvolver os algoritmos na língua portuguesa, se isso auxiliou ou não na compreensão da lógica de programação. Pode-se perceber que ajuda sim a compreender a construção dos códigos, sendo eles em português, conforme descreve o Estudante E, na sua resposta: *“Auxiliou sim, pois o português é o nosso dia a dia e a compreensão ao se ler um código que esteja em português fica mais viável, ainda mais para nós que somos iniciantes”*. O Estudante B também escreve que facilitou o entendimento dos códigos, *“[...] pois pode ser aprendido a lógica de forma mais fácil”*.

Dentre os estudantes que participaram da prática somente o Estudante A escreve que *“Para mim tanto faz, compreendo o inglês com facilidade”*, demonstrando que para alguns estudantes os *softwares* para desenvolvimento de programas serem na Língua Inglesa não é considerado uma barreira para a construção e aprendizagem dos algoritmos.

Ao analisar as respostas relativas a Questão 03 “Como você julga o seu aprendizado da lógica de programação? Ao ler o enunciado dos problemas você consegue facilmente criar o algoritmo? Explique.”, constatou-se que no geral os estudantes demonstraram que conseguem a partir do enunciado do problema criar o algoritmo para resolvê-lo. Uns com maior facilidade conforme descrito pelos Estudantes A e C, quando escrevem que: *“Compreendi bem o conteúdo passado e ao ler um enunciado dos problemas, consigo criar um algoritmo”* e *“Julgo desenvolvida, sim. Pois em português se torna mais fácil de entender”*, respectivamente, que poderiam desenvolver os algoritmos sem dificuldade.

Sendo que os demais estudantes apresentaram que conseguiriam criar os códigos, se não fossem complexos, extensos demais ou que necessitariam consultar o material de aula para esclarecer alguma dúvida. Esta percepção pode ser observada no comentário do Estudante B, quando este escreve que *“O aprendizado foi bom. Sim geralmente, mas necessito sempre consultar o material”*. Enquanto que o Estudante E diz que *“Dependendo do que for pedido é fácil, mas para criar algo mais extenso tenho dificuldades”*.

Já o Estudante F demonstrou ter um pouco de dificuldade para a criação de alguns algoritmos, dependendo da complexidade apresentada. Ele descreve que

“Dependendo o algoritmo tenho dificuldade em construí-lo, pois ainda não consegui um bom nível de conhecimento com esse tipo de lógica”. É comum alguns estudantes não compreenderem a lógica de programação no início, pois são vários fatores que influenciam no aprendizado necessário para a criação dos algoritmos. Souza, França e Lins (2014, p. 1575) descrevem que “Esta dificuldade deve-se a uma forte carga de conceitos abstratos que permeiam todo o conhecimento em programação”. Com isso, para alguns estudantes é necessário que sejam realizadas mais explicações ou exercícios, para que possam ser sanadas as dúvidas e auxiliar nas dificuldades apresentadas.

Com a Questão 04 - “O uso do *Software VisuAlg* auxiliou ou não na compreensão e criação dos algoritmos? Explique por que.”, pretendeu-se que os estudantes escrevessem as suas percepções sobre o uso do *VisuAlg*. Analisando as respostas, como um todo, percebe-se que todos julgaram o aprendizado dos algoritmos satisfatório com este *software*. Eles destacaram que pode ser observado o funcionamento do que foi desenvolvido, encontrar os erros com facilidade por meio das mensagens que são apresentadas, indicando qual o problema e algumas sugestões para correção. Além de “rodar” passo a passo o algoritmo, acompanhar o seu funcionamento e os valores que são salvos nas variáveis.

Percebe-se estas facilidades e benefícios apresentados pelos estudantes, ao analisar as respostas escritas pelo Estudante F quando ele escreve: *“Sim auxiliou muito, porque você faz algo e testa, se encontrar algum problema o software te mostra você corrige. Além de poder ver passo a passo o andamento do algoritmo”*, e pelo Estudante E *“Auxiliou, pois ele informa o erro e muitas vezes já dá a resposta para solucionar o mesmo”*. Todos os estudantes confirmaram que o uso do *VisuAlg* auxiliou de alguma forma para que eles pudessem compreender melhor a criação do código, realizar os testes destes e encontrar os possíveis erros.

Valente (1993) defende que existem basicamente dois tipos de utilização dos *softwares*, sendo que de um lado o estudante aprende por meio do computador e do outro o estudante “ensina” o computador como fazer o que ele quer, conforme já apresentado na seção 2.1. Com isso, baseando-se no que foi exposto por Valente (1993, p. 3) “[...] o computador pode ser visto como uma ferramenta que permite ao aprendiz resolver problemas ou realizar tarefas [...]”, pode-se classificar o *VisuAlg*

como um *software* que o estudante cria o seu algoritmo instruindo ao computador o que precisa ser feito, e considerá-lo como uma “ferramenta educacional”.

Em algumas questões as respostas dos estudantes foram semelhantes a outras que já haviam respondido, como nas Questões 05 e 06, que tratavam sobre os problemas para resolver as atividades e exercícios propostos, e as dificuldades para aprender os códigos e para resolver os algoritmos. A maioria dos estudantes escreveu que não teve grandes dificuldades para aprender e desenvolver o que foi proposto durante a prática pedagógica, dando-se destaque aos comentários dos Estudantes C e F que escreveram: “*A minha dificuldade foi mais da minha parte sendo ela a lógica de como fazer o algoritmo*” e “*A parte um pouco mais complicada é quando exige repetições de ações*”, respectivamente.

Para analisar melhor a compreensão dos estudantes sobre a criação dos algoritmos, foram propostas perguntas mais focadas no entendimento dos códigos e processos utilizados no desenvolvimento dos algoritmos. Com a Questão 07 - “Explique o que fazem os seguintes comandos: LEIA, SE, PARA.”, procurou-se saber se os estudantes haviam compreendido os conceitos e funcionalidade destes comandos. Dos seis estudantes participantes cinco escreveram, com suas palavras, o que eles compreenderam com os comandos apresentados e a sua função. Somente o Estudante C que descreveu com poucas palavras os termos da questão. Ele escreveu “*LEIA = salva; SE = compara*”. A lógica do Estudante com as comparações feitas está correta, só não pode-se ter certeza se ele esqueceu de citar o PARA ou não sabia o que ele faz.

Além de conceituar os comandos apresentados na Questão 07 e descrever o seu funcionamento, foi solicitado, na Questão 08 - “Explique o que faz o trecho do algoritmo a seguir”, que os estudantes analisassem o trecho código apresentado na Figura 26 e explicassem o que ele faz. Os estudantes somente observaram visualmente o algoritmo apresentado, sendo que eles não tinham disponível o *VisuAlg* para testá-lo.

Figura 26 - Questão 08, Questionário - Avaliação da Prática Pedagógica.

```

teste <- "S"

ENQUANTO (teste = "S") FACA
    ESCREVA("Digite um número: ")
    LEIA(num)
    SE (num MOD 2 = 0) ENTAO
        teste <- "S"
    SENAO
        teste <- "N"
    FIMSE
FIMENQUANTO

```

Fonte: O Autor (2016), a partir do *Software VisuAlg*.

Com a análise das respostas da Questão 08 percebe-se que todos os estudantes, que estavam presentes nesta aula, compreenderam o funcionamento do algoritmo apresentado nesta questão. Ao observar a resposta do Estudante E quando diz que *“Enquanto o número digitado for par o algoritmo continua pedindo para digitar um número, caso contrário se encerra”*, descrevendo exatamente o que o trecho de código faz. Como, também, explica o Estudante A quando escreve: *“Tenta se um número é par ou ímpar, e se for par ele pede outro número, mas se for ímpar ele encerra o código”*.

Questões como estas auxiliam o professor a perceber se os estudantes estão compreendendo como criar e analisar os algoritmos que já estão prontos, sendo este um requisito importante para a realização da correção dos *softwares* desenvolvidos. Muitas vezes o programador assume projetos que já estão em andamento e precisam analisar e compreender a lógica do programa, para fazer manutenções e complementos ao sistema.

Ao continuar com a análise das respostas dos estudantes ao Questionário - Avaliação da Prática Pedagógica, as duas últimas questões trataram sobre como os estudantes avaliam as aulas ministradas, e principalmente a criação dos algoritmos em português com o uso do *VisuAlg*. Para isso, com a Questão 09 - *“Você acharia interessante que as aulas de algoritmos sempre utilizassem algum software em português para criar os códigos? Justifique sua resposta.”*, procurou-se conhecer a opinião e percepção dos estudantes sobre a metodologia de ensino proposta e utilizada nas aulas.

Observa-se que os estudantes se mostraram satisfeitos com a metodologia utilizada e que consideram que o uso de *softwares* para aprender a criação de algoritmos pode auxiliar no ensino da lógica de programação, utilizando códigos em português. Isso comprova-se na resposta do Estudante B, que escreve que *“Sim, pois a maioria dos alunos aprenderia melhor assim”*. E quando o Estudante E destaca as potencialidades do uso de *softwares* para encontrar os erros em seus algoritmos (*“Acho essencial, pois com um software podemos aprender com nossos próprios erros”*), enfatizando o que Valente (1993) descreve em seus estudos quando compara os dois tipos de uso dos *softwares* (seção 2.1).

O Estudante A acha interessante utilizar *softwares* em português nas aulas iniciais de Algoritmos e Programação, mas apresenta que é necessário aprender em inglês também (*“Seria bom às vezes, porém acho necessário aprender em inglês”*). Este mesmo estudante escreveu em sua resposta para a Questão 02, que ele compreendia inglês com facilidade, com isso, a língua não seria um empecilho para a sua aprendizagem. Para complementar, vale ressaltar que o ensino de algoritmos na Língua Portuguesa é recomendado principalmente no início das aulas, para que todos os estudantes compreendam a lógica de programação necessária para criar os códigos. Posteriormente será utilizada alguma linguagem de programação que é usualmente em inglês.

Outro ponto importante que é interessante ser registrado se refere às respostas da Questão 10, *“Avalie esta prática pedagógica descrevendo o que poderia ter sido melhor.”*, quando os estudantes escrevem que se sentem satisfeitos com a prática pedagógica, conforme apresenta o Estudante C: *“[...] as aulas muito bem explicadas, conteúdo muito bem elaborado. Parabéns!!!!”*; e o Estudante E: *“Achei bem interessante aprender os algoritmos em português, o modo que as aulas foram planejadas também foram bem elaboradas, pois conseguimos desenvolver vários algoritmos a cada novo comando que aprendíamos”*.

Os Estudantes A e F apresentam que seria importante ter mais textos e videoaulas que orientem os estudantes durante a criação dos algoritmos, e que poderiam haver mais aulas para que os estudantes pudessem praticar mais. Esta percepção dos estudantes pode estar associada às aulas da prática serem em dias

consecutivos, com isso eles não tinham muito tempo para praticar em casa as atividades e exercícios propostos em aula.

No geral, analisou-se que praticamente todos os estudantes avaliaram positivamente a utilização do *Software VisuAlg* como ferramenta para o ensino da lógica de programação, por meio da criação de algoritmos escritos em Língua Portuguesa. Almeida et al. (2002) apresentam que com o uso de *softwares* pode-se constatar que:

A motivação dos alunos aumentou sensivelmente, uma vez que as aulas em laboratório, acrescida da possibilidade de escrever os programas/algoritmos diretamente no computador e da facilidade de testar o seu funcionamento, são bem mais estimulantes e ilustrativas do que na forma tradicional, onde os algoritmos são escritos no papel (ALMEIDA et al., 2002, p. 9).

Também se teve esta percepção, pois os estudantes se mostraram interessados e motivados na resolução dos exercícios propostos, ao implementar códigos mais elaborados e criando funcionalidades extras para melhorarem os seus algoritmos. Percebeu-se, também, que todos julgaram válida a utilização do *VisuAlg* para a criação dos algoritmos em português, principalmente pela possibilidade de poderem testar e visualizar o resultado ou os erros do código, visualizando as dicas e sugestões apresentadas. Isso facilitou a resolução dos problemas computacionais apresentados nos exercícios propostos durante a prática pedagógica.

5 CONSIDERAÇÕES FINAIS

Esta pesquisa foi desenvolvida a partir da prática pedagógica realizada com o curso Técnico em Informática, na disciplina de Algoritmos, do Módulo III - Assistente em Programação, da Escola Estadual de Educação Profissional Estrela. Seu objetivo principal foi: Analisar as potencialidades do *Software VisuAlg* como recurso didático nas aulas da disciplina de Algoritmos, do curso Técnico em Informática, da Escola Estadual de Educação Profissional Estrela.

Um dos principais estímulos para pesquisar sobre este tema surgiu ao analisar as aulas iniciais de algoritmos e programação ministradas ao curso Técnico em Informática, quando percebeu-se que alguns estudantes demoravam mais para entender a lógica de programação necessária para resolver os problemas computacionais propostos. Junior, Vieira e Vieira (2015) destacam, também, que certos estudantes apresentam dificuldade de raciocínio lógico na construção dos códigos, na capacidade de abstração e na leitura e interpretação dos textos dos enunciados. Além do exposto, acredita-se que outro motivo para a dificuldade em aprender a programar está relacionado à compreensão dos comandos em Inglês, pois sabe-se que alguns estudantes podem não compreender o significado de certos termos nesta língua.

Com o exposto, diversos autores e professores vêm propondo que sejam utilizadas e criadas ferramentas computacionais para auxiliarem no ensino da lógica de programação, de maneira que os estudantes possam compreender melhor o processo de criação dos algoritmos. Souza (2009, p. 8) indica que devido “as dificuldades apresentadas” o ensino convencional utilizado até algum tempo atrás

“deve ser deixado de lado” e que o “uso de ferramentas de auxílio ao aprendizado deve ser estudado e difundido para que o rendimento dos alunos seja melhorado”.

A escolha pelo *VisuAlg* justifica-se por tratar-se de um software desenvolvido e pensado para o ensino de algoritmos, apresenta diferentes recursos educacionais que podem ser utilizados como auxiliares na hora da criação dos códigos. Além disso, os autores Almeida (2013), Costa (2010), Guedes (2014), Leite et al. (2013) e Souza (2009) descrevem experiências e vantagens na utilização do *VisuAlg*, indicando que é *software* muito usado por diferentes professores em várias instituições de ensino.

Em vista disso, surgiu a questão: Como o *Software VisuAlg* pode ser utilizado de forma que seja um potente recurso didático para o ensino da lógica de programação na disciplina de Algoritmos?

Para respondê-la, foram estabelecidos três objetivos específicos, são eles: identificar os conhecimentos dos estudantes em relação à programação; desenvolver uma proposta pedagógica, com estudantes do Módulo III do curso Técnico em Informática, que envolva a utilização do *Software VisuAlg* para a construção de algoritmos; analisar como a utilização do *Software VisuAlg* contribui para a aprendizagem da lógica de programação.

Após a exploração da prática pedagógica os dados coletados foram divididos em categorias e analisados com aproximações à Análise Textual Discursiva, conforme propõem Moraes e Galiazzi (2013), sendo que cada categoria surgiu para atingir os objetos específicos apresentados.

Na Categoria 1 - “Conhecimentos sobre algoritmos” foram analisadas as respostas dos estudantes ao Questionário de Sondagem (APÊNDICE D), onde constatou-se que: 1) Alguns dos estudantes que participaram da pesquisa apresentavam conhecimentos gerais sobre o que é um problema computacional e a ideia de lógica; 2) A maioria dos estudantes não conhecia os conceitos e algoritmos e os comandos que são utilizados na criação dos códigos em português; 3) Dentre os sete participantes da prática pedagógica somente um estudante descreveu que não conhecia o nome de nenhuma linguagem de programação.

A Categoria 2 - “Contribuições do *Software VisuAlg* no ensino de algoritmos” constatou-se que os estudantes compreenderam os comandos e algoritmos trabalhados, pois conseguiram analisar os problemas apresentados nos exercícios e criar a codificação necessária para resolvê-los. Em certas atividades alguns estudantes criaram formas diferentes de resolução e propondo algoritmos extras, demonstrando que estavam compreendendo a lógica necessária para o início da programação.

Além disso, pode-se destacar: 1) Com a apresentação das três formas de representação dos algoritmos trabalhadas (Descrição Narrativa, Fluxograma e Português Estruturado) os estudantes puderam compreender o processo de análise do problema apresentado até a criação do algoritmo; 2) Os estudantes desenvolveram novos algoritmos para simular situações e brincadeiras, demonstrando que estavam compreendendo a utilização do *VisuAlg* para a criação dos códigos; 3) A utilização do *VisuAlg* se mostrou muito útil na criação dos algoritmos, os estudantes puderam testar o código digitado encontrando facilmente os erros apresentados, pois o *software* mostra o local no qual está o problema e dicas de como resolvê-lo.

Souza (2009, p. 8-9) descreve que o uso do *VisuAlg*:

[...] nos estágios iniciais do ensino de programação tem-se mostrado bastante produtivo por permitir que desde o início os estudantes tenham contato com um ambiente de desenvolvimento próximo ao que encontrarão em sua vida profissional, embora mais simples e usando uma linguagem de programação sem tantos recursos. Ambientes como o *VisuAlg* promovem a experimentação e permitem desde cedo o entendimento do funcionamento de um programa de computador.

Na Categoria 3 - “Percepções dos estudantes sobre o *Software VisuAlg*” foi realizada a análise do Questionário - Avaliação da Prática Pedagógica (APÊNDICE O), na qual os estudantes puderam apresentar as suas percepções, os comentários e demonstrar quais as dificuldades encontradas durante a prática pedagógica. As questões trataram de assuntos como: qual a forma de ensino de algoritmos os estudantes aprenderam mais (digitando no editor de texto ou utilizando o *Software VisuAlg*); se aprender algoritmos em português auxiliou no aprendizado dos códigos; quais as potencialidades do uso do *VisuAlg* no ponto de vista dos estudantes; e,

analisar a compreensão dos estudantes sobre algoritmos prontos e de alguns comandos.

As respostas dos estudantes expressam que: 1) Os estudantes preferiram desenvolver os algoritmos utilizando o *Software VisuAlg* ao editor de textos, devido a facilidade de poder testar e encontrar os erros; 2) Aprender a programar em português permitiu que os estudantes focassem na resolução dos problemas computacionais e na lógica necessária para resolvê-los, não em compreender o significados de comandos em inglês das linguagens de programação; 3) Foram destacadas as funcionalidades do *VisuAlg* como ponto forte na sua utilização, pois pode-se visualizar por meio da indentação qual comando está dentro de outro. E identificar os erros apresentados nas mensagens com dicas de como resolvê-los; 4) Todos os estudantes que responderam ao questionário demonstraram saber qual o resultado do algoritmo de exemplo apresentado e qual a função dos comandos deste; 5) Os estudantes demonstraram que a prática pedagógica foi satisfatória e que eles puderam compreender e criar os algoritmos propostos sem muita dificuldade, recomendando o seu uso em outras turmas.

Com a experiência adquirida e os relatos apresentados pelos estudantes na prática pedagógica, pode-se perceber que, conforme também apresentado pelos autores Souza (2009), Costa (2010), Almeida (2013), Leite et al. (2013), Guedes (2014), Junior, Vieira e Vieira (2015) e Puga e Rissetti (2016), a utilização de *softwares* e ferramentas auxilia no ensino de algoritmos e nas aulas iniciais de programação. Souza (2009, p. 8) descreve que o uso de diferentes recursos no ensino da lógica de programação “[...] facilita o entendimento de como um programa de computador funciona, possibilita um *feedback* imediato sobre a correção e exatidão do pseudocódigo digitado” além de instigar o estudante a experimentar e testar o algoritmo que ele está desenvolvendo, para ver o seu resultado e acompanhar as alterações nos conteúdos das variáveis no decorrer da depuração do código.

Além disso, Almeida et al. (2002) destacam que a motivação dos estudantes aumenta, pois eles podem comprovar que os códigos digitados estão funcionando, com a facilidade de encontrar os erros que surgirem durante o seu funcionamento. Almeida et al. (2002, p. 9) ainda descrevem que utilizar ferramentas e *softwares* para

a criação de algoritmos é “[...] bem mais estimulante e ilustrativa do que na forma tradicional, onde os algoritmos são escritos no papel”.

Ao concluir a pesquisa, pode-se perceber que, conforme citado por diferentes autores e professores em seus artigos, livros e relatos, o uso de ferramentas que possam auxiliar no ensino e aprendizagem da lógica necessária para a criação de algoritmos e na programação é bem vinda em sala de aula. Pois proporciona aos estudantes maneiras de aprender, que o estimulem a conhecer cada vez mais sobre o desenvolvimento do código-fonte de programas, compreendam a lógica envolvida na sua criação e desenvolvam habilidades que o habilitem trabalhar na área de Informática e Computação.

Como continuidade deste estudo, recomenda-se que outras práticas sejam realizadas, a fim de compreender as implicações e contribuições pedagógicas que novas técnicas e ferramentas computacionais possam trazer para o ensino da lógica de programação, nos diferentes níveis de ensino.

REFERÊNCIAS

- ALMEIDA, Eliana S. de; COSTA, Evandro de B.; SILVA, Klebson dos S.; PAES, Rodrigo de B.; ALMEIDA, André Atanasio M.; BRAGA, Julian D. Herrera. **AMBAP: Um Ambiente de Apoio ao Aprendizado de Programação**. 2002. Departamento de Tecnologia da Informação, Universidade Federal de Alagoas (TCI/UFAL). Disponível em: <<http://www.lbd.dcc.ufmg.br/colecoes/wei/2002/006.pdf>>. Acesso em: 19 ago. 2017.
- ALMEIDA, Rafael Soares de. **Aprendendo Algoritmo com VisuAlg**. Rio de Janeiro: Editora Ciência Moderna Ltda., 2013.
- APOIO, Informática Ltda. **VisuAlg**. 2016. Disponível em: <<http://www.apoioinformatica.inf.br/produtos/visualg>>. Acesso em: 13 mar. 2016.
- BERG, Alexandre Cruz; FIGUEIRÓ, Joice Pavex. **Lógica de Programação**. 2ª ed. Canoas, Rio Grande do Sul: Ulbra, 1998.
- BRASIL, Casa Civil. **Proteção da propriedade intelectual de programa de computador**. LEI Nº 9.609, DE 19 DE FEVEREIRO DE 1998. Disponível em: <http://www.planalto.gov.br/ccivil_03/Leis/L9609.htm>. Acesso em: 18 out. 2015.
- CARVALHO, Flávia Pereira de. **Apostila de Lógica de Programação - ALGORITMOS**. FIT - Faculdade de Informática de Taquara, Curso de Sistemas de Informação, 2007. Disponível em: <https://fit.faccat.br/~fpereira/apostilas/apostila_algoritmos_mar2007.pdf>. Acesso em: 17 out. 2015.
- CELESTINO, André Luis. **O limite entre o Desenvolvedor e o Usuário**. 2015. Disponível em: <<https://www.profissionaisiti.com.br/2015/03/o-limite-entre-o-desenvolvedor-e-o-usuario/>>. Acesso em: 21 abr. 2017.
- COSTA, R. H. P.; FARIA, E. S. J.; YAMANAKA, K. **Programação em Duplas no Aprendizado de Programação de Computadores em um Curso de Engenharia de Produção: Um Estudo Empírico**. REIC. Revista Eletrônica de Iniciação Científica (on-line), v. X, p. 4, 2010.

DRJAVA TEAM. **About DrJava**. Disponível em: <<http://www.drjava.org/>>. Acesso em: 15 out. 2017.

EEEPE, Escola Estadual de Educação Profissional Estrela. **Projeto Político Pedagógico**. Estrela, 2015.

FABRI, José Augusto. **Ferramentas que auxiliam o ensino de algoritmos**. Fundação Educacional do Município de Assis - Faculdade de Tecnologia de Ourinhos, 2008. Disponível em: <<https://engenhariasoftware.wordpress.com/2008/07/15/>>. Acesso em: 19 mar. 2016.

FERRAZ, Ana Paula do C. M.; BELHOT, Renato Vairo. Taxonomia de Bloom: revisão teórica e apresentação das adequações do instrumento para definição de objetivos instrucionai. **Revista Gestão & Produção**. vol.17 no. 2 São Carlos, 2010. Disponível em: <<http://dx.doi.org/10.1590/S0104-530X2010000200015>>. Acesso em: 15 out. 2017.

FORBELLONE, André Luiz Villar; EBERSPÄCHER, Henri Frederico. **Lógica de programação: a construção de algoritmos e estruturas de dados**. 3ª ed. São Paulo: Prentice Hall, 2005.

GOMES, Anabela de Jesus; MENDES, António José. À procura de um contexto para apoiar a aprendizagem inicial de programação. **Revista Educação, Formação & Tecnologias** (janeiro-junho, 2015), 8 (1),13-27. Disponível em: <<http://eft.educom.pt/index.php/eft/article/viewFile/439/212>>. Acesso em: 26 mai. 2016.

GUEDES, Sérgio. **Lógica de programação algorítmica**. São Paulo: Pearson Education do Brasil (Biblioteca Universitária Pearson), 2014.

HOSTINS, Higor; RAABE, André. **Auxiliando a aprendizagem de algoritmos com a ferramenta Webportugol**. XXVII Congresso da SBC 2007, Rio de Janeiro, RJ. Disponível em: <<http://www.lbd.dcc.ufmg.br/colecoes/wei/2007/0010.pdf>>. Acesso em: 26 mai. 2016.

ISO 5807:1985. **Information processing** - Documentation symbols and conventions for data, program and system flowcharts, program network charts and system resources charts. 2005. Disponível em: <<https://www.iso.org/standard/11955.html>>. Acesso em: 15 out. 2017.

JUNIOR, José Augusto Teixeira de Lima; VIEIRA, Carlos Eduardo Costa; VIEIRA, Priscila de Paula. **Dificuldades no processo de aprendizagem de Algoritmos: uma análise dos resultados na disciplina de AL1 do Curso de Sistemas de Informação da FAETERJ - Campus Paracambi**. Cadernos UniFOA - Centro Universitário de Volta Redonda. Edição 27, Abril de 2015. Disponível em: <<http://web.unifoa.edu.br/cadernos/edicao/27/5-15.pdf>>. Acesso em: 14 mai. 2016.

KÜFFER, Teresa. **Os hiperespaços para a educação formal, não formal e informal**. 2011. Disponível em: <<http://www.sophia.org/tutorials/os-hiperespacos-para-a-educacao-formal-nao-formal>>. Acesso em: 20 mar. 2016.

LEITE, Vanessa Matias; SENEFONTE, Helen C. M.; BARBOSA, Cinthyan R.S.C.; SEABRA, Rodrigo Duarte. *VisuAlg*: Estudo de Caso e Análise de Compilador destinado ao ensino de Programação. **Revista Nuevas Ideas en Informática Educativa TISE 2013**. Disponível em: <<http://www.tise.cl/volumen9/TISE2013/637-640.pdf>>. Acesso em: 26 mai. 2016.

MARTINS, Gilberto Andrade. Estudo de caso: uma reflexão sobre a aplicabilidade em pesquisas no Brasil. **RCO - Revista de Contabilidade e Organizações - FEARP/USP**, v. 2, n. 2, p. 8 - 18 jan./abr. 2008. Disponível em: <<http://www.revistas.usp.br/rco/article/viewFile/34702/37440>>. Acesso em: 05 jun. 2016.

MICHAELIS. **Dicionário de Português Online**. 2009 - Editora Melhoramentos Ltda. Disponível em: <<http://michaelis.uol.com.br/>>. Acesso em: 08 fev. 2016.

MORAES, Roque; GALIAZZI, Maria do Carmo. **Análise textual discursiva**. 2 ed. rev. - Ijuí Ed. Unijuí, 2013.

MORAES, Paulo Sérgio de. **Lógica de Programação**. Unicamp - Centro de Computação (DSC), 2000. Disponível em: <<http://www.inf.ufsc.br/~vania/teaching/ine5231/Logica.pdf>>. Acesso em: 18 out. 2015.

MORAIS, Carlos; FRANCO, Sérgio Roberto K. Avaliação de Alunos de Turmas Heterogêneas no Ensino a Distância. **Anais do VI Congresso Ibero-americano de Telemática (CITA 2011)** - Gramado RS (Brasil), 16-18 Maio 2011. Disponível em: <<http://seer.ufrgs.br/cadernosdeinformatica/article/view/v6n1p155-162>>. Acesso em: 08 jul. 2017.

MORAN, José M.; MASETTO, Marcos T.; BEHRENS, Marinda A. **Novas Tecnologias e Mediação Pedagógica**. 13 ed. - Campinas, SP: Papirus, 2007.

MOREIRA, Marco Antônio. **Aprendizagem Significativa**: a teoria e textos complementares. São Paulo: Editora Livraria da Física, 2011.

MOREIRA, Jonas V.; YNOGUTTI, Carlos A. Piratas do Futuro - Ferramenta para suporte ao ensino de algoritmos e lógica de programação. **Anais do Congresso de Iniciação Científica do INATEL - INCITEL 2012**, p. 104-113. Disponível em: <<http://www.inatel.br/biblioteca/artigos-cientificos/2012/5890-piratas-do-futuro-ferramenta-para-suporte-ao-ensino-de-algoritmos-e-logica-de-programacao/>>. Acesso em: 08 out. 2017.

MOURA, Isabel C. **Visualizing the Execution of Programming Worked-out Examples with Portugol**. Proceedings of the World Congress on Engineering 2013, Vol I, WCE 2013, July 3-5, 2013, London, U.K. Disponível em: <http://www.iaeng.org/publication/WCE2013/WCE2013_pp404-408.pdf>. Acesso em: 26 mai. 2016.

NETO, Antônio G. S. **Java na Web**. Rio de Janeiro: Editora Ciência Moderna, 2011.

PIMENTEL, Edson. **Teste de mesa**. Centro de Matemática, Computação e Cognição - Universidade Federal do ABC, 2015. Disponível em: <http://tidia-ae.ufabc.edu.br/access/content/group/9ad27a50-bb10-46c6-bc12-c7a73ddec6cd/_PDFs_Aulas/AULA04/PI-AULA04-P01-TestedeMesa.pdf>. Acesso em: 02 nov. 2015.

PUGA, Sandra; RISSETI, Gerson. **Lógica de programação e estruturas de dados, como aplicações em Java**. 2. ed. - São Paulo: Pearson Prentice Hall, 2009.

_____. **Lógica de programação e estruturas de dados, como aplicações em Java**. 3. ed. - São Paulo: Pearson Education do Brasil, 2016.

RICHARDSON, R. J. **Pesquisa Social: métodos e técnicas**. São Paulo: Atlas, 1999.

SCHNEIDER, Henrique Nou. **A educação na contemporaneidade: flexibilidade, comunicação e colaboração**. Int. J. Knowl. Eng. Manage., ISSN 2316-6517, Florianópolis, v. 2, n. 2, p. 86-104, mar./mai., 2013. Disponível em: <<http://incubadora.periodicos.ufsc.br/index.php/IJKEM/article/viewFile/2140/2472>>. Acesso em: 14 fev. 2016.

SIGNIFICADOS. **Significado de Desktop**. Disponível em: <<http://www.significados.com.br/desktop/>>. Acesso em: 08 out. 2017.

SOUZA, Cláudio Morgado de. **VisuAlg - Ferramenta de Apoio ao Ensino de Programação**. Universidade Severino Sombra, CECETEN. Revista TECCEN, volume 2, número 2, setembro, ISSN 1984-0993, 2009.

SOUZA, Márcia Valéria Rocha de; FRANÇA, César; LINS, Walquíria. **Ferramentas de apoio ao aprendizado de programação na FAFICA**. XXXIV Congresso da Sociedade Brasileira de Computação - CSBC 2014. Disponível em: <<http://www.lbd.dcc.ufmg.br/colecoes/wei/2014/0027.pdf>>. Acesso em: 27 mai. 2016.

TONET, Bruno. **Software Visualg 2.0**. NAPRO - Núcleo de Apoio Aprendizagem de Programação. Universidade de Caxias do Sul. Disponível em: <<http://www.dainf.ct.utfpr.edu.br/~eteocles/visualg.pdf>>. Acesso em: 08 out. 2017.

VALENTE, José Armando. **Diferentes usos do Computador na Educação**. Núcleo de Informática Aplicada à Educação - NIED/UNICAMP, 1993. Disponível em: <<http://ffalm.br/gied/site/artigos/diferentesusoscomputador.pdf>>. Acesso em: 20 set. 2015.

_____. **O computador na sociedade do conhecimento.** NIED - Universidade Estadual de Campinas. Campinas, SP: Unicamp/NIED, 1999.

YIN, Robert K. **Estudo de caso: planejamento e métodos.** 5. ed. Porto Alegre: Bookman, 2015.

APÊNDICES

APÊNDICE A - Termo de Concordância da Direção da Instituição de Ensino

A senhora Diretora

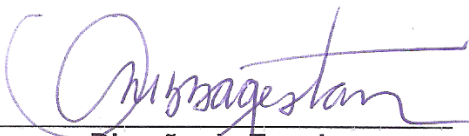
Eu, Fabrício Hartmann Borba, estudante regularmente matriculado no Curso de Pós-graduação *Stricto Sensu*, Mestrado em Ensino do Centro Universitário UNIVATES de Lajeado/RS, venho solicitar a autorização para coletar dados neste estabelecimento de ensino, para realização de minha pesquisa de Mestrado, intitulada: "O *SOFTWARE VISUALG* COMO RECURSO DIDÁTICO NO DESENVOLVIMENTO DA LÓGICA DE PROGRAMAÇÃO". O objetivo geral desta investigação é "Analisar as contribuições do *Software VisuAlg* como recurso didático nas aulas da disciplina de Algoritmos, do curso Técnico em Informática.

Afirmo ainda, que as coletas de dados serão realizadas por meio da aplicação de uma prática pedagógica, observações, questionários, fotografias, entrevistas e testes aos alunos da referida disciplina.


Desde já, agradeço a disposição, visto que a pesquisa contribuirá para o desenvolvimento do ensino da lógica de programação.

Pelo presente termo de concordância declaro que autorizo a realização da pesquisa prevista.

Data: 18 / 08 / 2016



Direção da Escola
Ana Rita Berti Bagestan
Diretora - Id. Func.: 1302027/02
E.E.E. Profissional de Estrela
D.O. 20/01/16 - Pág.: 21



Fabrício Hartmann Borba
Mestrando em Ensino - UNIVATES

APÊNDICE B - Carta de Anuência



Escola Estadual de Educação
Profissional Estrela

CARTA DE ANUÊNCIA

Aceito que o pesquisador Fabrício Hartmann Borba, pertencente ao Programa de Pós-Graduação - Mestrado em Ensino desenvolva sua pesquisa intitulada “O *Software VisuAlg* como recurso didático no ensino da lógica de programação”, sob a orientação da Professora Dra. Miriam Ines Marchi e coorientação da Professora Dra. Márcia Jussara Hepp Rehfeldt, vinculada ao **CETEC** - Centro de Ciências Exatas e Tecnológicas, podendo citar o nome desta Instituição em suas publicações (dissertação, artigos e eventos científicos).

Ciente dos objetivos, métodos e técnicas que serão utilizados nessa pesquisa, concordo em fornecer todos os subsídios para seu desenvolvimento, desde que seja assegurado o que segue:

- 1) A garantia de solicitar e receber esclarecimentos antes, durante e depois do desenvolvimento da pesquisa.
- 2) Que não haverá nenhuma despesa para esta instituição que seja decorrente da participação nessa pesquisa.
- 3) No caso do não cumprimento dos itens acima, a liberdade de retirar minha anuência a qualquer momento da pesquisa sem penalização alguma.

Estrela, 18 de agosto de 2016


Ana Rita Berti Bagestan
Diretora - Id. Func.: 1302027/02
E.E.E. Profissional de Estrela
D. O. 20/01/16 - Pág.: 21

Escola Estadual de Educação Profissional Estrela
Diretora Ana Rita Berti Bagestan
Endereço: Rua Coronel Mussnich, 270
Estrela/RS
Contatos: (51) 3712 1185 – direcao@eeepe.com.br
www.eeepe.com.br

APÊNDICE C - Termo de Consentimento Livre Esclarecido

Com o intuito de alcançar o objetivo proposto para este projeto: “O *Software VisuAlg* como recurso didático no ensino da lógica de programação”, venho convidar-lhe a participar desta pesquisa que faz parte da dissertação de mestrado desenvolvida no programa de Pós-graduação *Stricto Sensu*, Mestrado em Ensino, tendo como Orientadora a Professora Miriam Ines Marchi e como Coorientadora a Professora Márcia Jussara Hepp Rehfeldt.

Deste modo, no caso de concordância em participar desta pesquisa ou deixar participar (estudantes menores), ficará ciente de que a partir da presente data:

- os direitos da entrevista gravada ou respondidas (questionários) realizado pelo pesquisador, será utilizada integral ou parcialmente, sem restrições;
- estará assegurado o anonimato nos resultados dos dados obtidos, sendo que todos os registros ficarão de posse do pesquisador por cinco anos e após esse período serão extintos.

Será garantido também:

- receber a resposta e/ou esclarecimento de qualquer pergunta e dúvida a respeito da pesquisa;
- poderá retirar seu consentimento a qualquer momento, deixando de participar do estudo, sem que isso traga qualquer tipo de prejuízo.

Assim, mediante deste Termo de Consentimento Livre e Esclarecido, declaro que autorizo minha participação nesta pesquisa, por estar esclarecido e não me oferecer nenhum risco de qualquer natureza. Declaro ainda, que as informações fornecidas nesta pesquisa podem ser usadas e divulgadas neste curso de Pós-graduação *Stricto Sensu*, Mestrado em Ensino da Universidade do Vale do Taquari - Univates, bem como nos meios científicos, publicações eletrônicas e apresentações profissionais.

Participante da pesquisa e/ou responsável

Pesquisador: Fabrício Hartmann Borba
fabricao.borba@univates.br

Lajeado/RS, 18 de agosto de 2016.

APÊNDICE D - Conhecimentos em relação à programação

QUESTIONÁRIO - SONDAÇÃO SOBRE CONHECIMENTOS EM RELAÇÃO À PROGRAMAÇÃO

- 4) Qual a sua idade?
- 5) Qual o ano que você se formou no Ensino Médio?
- 6) Você trabalha na área de informática? O que você faz? Há quanto tempo?
- 7) Em sua opinião, o que é um problema?
- 8) Em sua opinião, o que é lógica?
- 9) O que você entende por algoritmo?
- 10) Marque quais dos termos a seguir você já ouviu falar:
 - () Tipos de dados.
 - () Variáveis.
 - () Português estruturado.
 - () Portugol.
 - () Comandos condicionais.
 - () Comandos de repetição.
 - () *VisuAlg*.
- 11) Já ouviu falar em linguagem de programação? Cite o nome das que conhece.
- 12) Escreva algum comando que conheça sobre programação.

APÊNDICE E - Plano da Aula 1

Conteúdos:

- Conceito de problema, lógica e algoritmo.
- Operadores aritméticos (**, *, /, +, -, div e mod).
- Tipos de dados primitivos (inteiro, real, caracter e lógico).
- Conceito de variáveis e constantes.
- Estrutura do algoritmo.

Objetivos:

- Conhecer o conceito de problema e lógica.
- Conhecer os principais operadores aritméticos.
- Conhecer e saber identificar os tipos de dados primitivos.
- Conhecer conceitos de variáveis e constantes.
- Conhecer e saber construir a estrutura de um algoritmo (descrição narrativa).

Atividades:

- Apresentar a proposta da prática pedagógica para a turma de estudantes.
- Assinatura do Termo de Consentimento Livre Esclarecido.
- Aplicar o questionário de sondagem (APÊNDICE D).
- Discutir os assuntos que serão trabalhados durante a prática pedagógica e analisar os conhecimentos dos estudantes sobre algoritmos.
- Discutir o conceito de problema, lógica e algoritmo.
- Apresentar e explicar os operadores aritméticos.
- Apresentar e explicar os tipos de dados primitivos por meio de explicações e exemplos.
- Apresentar e explicar o conceito de variáveis e constantes, apresentando exemplos.
- Apresentar e explicar a estrutura de construção de um algoritmo no formato de descrição narrativa.
- Aplicação e resolução de exercícios para fixação (APÊNDICE J).

Recursos:

- Aula expositiva dialogada.
- Caderno e lápis ou editor de texto.
- Laboratório de Informática.
- Lista de exercícios (APÊNDICE J).
- Quadro branco.
- Questionário de sondagem (APÊNDICE D).

APÊNDICE F - Plano da Aula 2

Conteúdos:

- Operadores:
 - o relacionais (>, >=, <, <=, =, <>);
 - o lógicos (E, OU e NÃO);
 - o atribuição (<-).
- Tipos de representações de algoritmos.
- Conceito pseudocódigo e português estruturado.
- *Software VisuAlg*.
- Criação de algoritmos em português estruturado.

Objetivos:

- Conhecer e saber aplicar os principais operadores relacionais, lógicos e de atribuição.
- Conhecer os tipos de representações de algoritmos.
- Conhecer o conceito de pseudocódigo e português estruturado.
- Conhecer a estrutura para a criação de um algoritmo, em português estruturado, com a utilização do *Software VisuAlg*.

Atividades:

- Explicar e exemplificar os principais operadores relacionais, lógicos e de atribuição.
- Apresentar e explicar o conceito de algoritmo, pseudocódigo e português estruturado.
- Apresentar e explicar a estrutura para a criação de um algoritmo, em português estruturado, com a utilização do *Software VisuAlg*.
- Aplicação e resolução de exercícios para fixação (APÊNDICE K).

Recursos:

- Aula expositiva dialogada.
- Laboratório de Informática.
- Lista de exercícios (APÊNDICE K).
- Projetor multimídia.
- Quadro branco.
- *Software VisuAlg*.

APÊNDICE G - Plano da Aula 3

Conteúdo:

- Comandos de entrada e saída de dados.
- Comando condicional SE (simples, composto e encadeado).
- Utilização do *Software VisuAlg* para elaborar e testar os algoritmos desenvolvidos.

Objetivos:

- Conhecer os comandos de entrada e saída de dados.
- Conhecer o comando condicional SE (simples, composto e encadeado).
- Conhecer a utilização e opções do *Software VisuAlg* para elaborar e testar os algoritmos desenvolvidos.
- Saber utilizar o *Software VisuAlg*.

Atividades:

- Apresentar e explicar os comandos de entrada e saída de dados.
- Apresentar e explicar o comando condicional SE (simples, composto e encadeado).
- Demonstrar a utilização e as diferentes opções do *Software VisuAlg* para elaborar e testar os algoritmos desenvolvidos.
- Aplicação e resolução de exercícios práticos no *VisuAlg* para fixação (APÊNDICE L).

Recursos:

- Aula expositiva dialogada.
- Laboratório de Informática.
- Lista de exercícios (APÊNDICE L).
- Projetor multimídia.
- Quadro branco.
- *Software VisuAlg*.

APÊNDICE H - Plano da Aula 4

Conteúdo:

- Comando de repetição PARA.

Objetivos:

- Conhecer o comando para repetição de código (PARA).

Atividades:

- Discutir sobre a utilização do *Software VisuAlg*.
- Esclarecer as dúvidas em relação ao *Software VisuAlg*.
- Apresentar o comando de repetição PARA, explicando o seu funcionamento e utilização.
- Testar a execução passo a passo do algoritmo criado no *Software VisuAlg*.
- Aplicação e resolução de exercícios práticos no *Software VisuAlg* (APÊNDICE M).

Recursos:

- Aula expositiva dialogada.
- Laboratório de Informática.
- Lista de exercícios práticos (APÊNDICE M).
- Projetor multimídia.
- Quadro branco.
- *Software VisuAlg*.

APÊNDICE I - Plano da Aula 5

Conteúdo:

- Comando de repetição ENQUANTO.

Objetivos:

- Conhecer o comando para repetição de código (ENQUANTO).
- Investigar a percepção dos estudantes sobre prática desenvolvida e os conhecimentos sobre algoritmos adquiridos.

Atividades:

- Apresentar o comando de repetição ENQUANTO, explicando o seu funcionamento e utilização.
- Aplicação e resolução de exercícios práticos no *VisuAlg* (APÊNDICE N).
- Discutir sobre as aulas e proposta pedagógica desenvolvida.
- Discutir sobre a escrita de algoritmos no caderno (editor de texto) em comparação com o uso do *Software VisuAlg*.
- Aplicar o questionário de avaliação das aulas (APÊNDICE O).

Recursos:

- Aula expositiva dialogada.
- Formulário *on-line* no *Google Drive* (APÊNDICE O).
- Laboratório de Informática.
- Lista de exercícios práticos (APÊNDICE N).
- Projetor multimídia.
- Quadro branco.
- *Software VisuAlg*.

APÊNDICE J - Lista de Exercícios: Tipos de dados, Variáveis e Algoritmos

RESOLVA OS EXERCÍCIOS ABAIXO SOBRE TIPOS DE DADOS, VARIÁVEIS E ALGORITMOS

1) Classifique os dados abaixo de acordo com seu tipo:

(I) Inteiro	(R) Real	(C) Caracter	(L) Lógico
(R) 3.14	() -54	() "false"	() "0."
(C) "tamanho"	() "+3456"	() 0.0	() '0'
() 30	() 'C'	() "12"	() true
() "inteiro"	() 12.69	() -16.65	() "Z"
() 55	() 0	() "c/d"	() 'Z'
() 1.27	() -16E10	() false	() "cinco"

2) Verifique se os nomes de variáveis a seguir estão corretos e justifique sua resposta:

- A. 3idade
- B. nome
- C. \$reais
- D. CEP
- E. num1
- F. telefone 1
- G. 1ºEndereço
- H. nome_completo
- I. media/total
- J. R\$

3) Desenvolva os algoritmos para as seguintes tarefas:

- A. Tomar um banho.
- B. Você é porteiro de uma festa e somente pessoas autorizadas podem entrar. Selecionar quem pode entrar.
- C. Calcular o volume de uma sala.
- D. Limpar e organizar o quarto.
- E. Ajustar a temperatura ideal da água do banho.

APÊNDICE K - Lista de Exercícios: Operadores de Atribuição, Lógicos e Relacionais e Algoritmos

RESOLVA OS EXERCÍCIOS ABAIXO SOBRE OPERADORES DE ATRIBUIÇÃO, LÓGICOS E RELACIONAIS E ALGORITMOS

1) Resolver as expressões a seguir, considerando que:

$a = V, b = V, c = V, d = F, e = F$

- a. $(a \text{ OU } b)$
- b. $(d \text{ OU } e)$
- c. $(c \text{ OU } e \text{ OU } a)$
- d. $(d \text{ OU } e \text{ E } a)$
- e. $(!a \text{ OU } !c) \text{ E } !d$
- f. $(!d \text{ OU } !b \text{ E } a)$
- g. $(!d \text{ OU } !e) \text{ E } (!a \text{ E } !e)$
- h. $(d \text{ OU } e) \text{ OU } (a \text{ OU } b) \text{ E } !c$
- i. $!((a \text{ OU } b \text{ OU } c) \text{ E } (d \text{ OU } e))$
- j. $(!d \text{ OU } a)$
- k. $!(! (a \text{ E } c) \text{ E } !(e \text{ E } d))$

2) Resolva as expressões e descubra se o resultado é Verdadeiro ou Falso, considerando que:

$a = 15, b = 8, c = 53, d = 10, e = 48, f = 24, g = 2$

- A. $(a * b) \geq 99$
- B. $(f * e < 150) \text{ OU } (c + c \leq 100)$
- C. $(b * b = 16) \text{ E } (d * e = 480)$
- D. $(100 - 52 <> e)$
- E. $(b * 10) \geq 16$
- F. $(24 * f) = 288$
- G. $((c * b) + 55) \geq 130$
- H. $(c * d = 530) \text{ OU } (f * 2 = 48) \text{ E } (15 * a = 225)$

3) Desenvolva os algoritmos solicitados abaixo utilizando o *Software VisuAlg*:

- a) Faça um algoritmo que leia dois números inteiros e calcule a soma.
- b) Faça um algoritmo que leia dois números inteiros, nas variáveis X e Y. Troque os números entre as variáveis e escreva na tela.
- c) Faça um algoritmo que solicite o nome do usuário e a sua idade, no final escreva o nome e a idade da pessoa.
- d) Faça um algoritmo que receba 3 notas de um estudante e escreva a média no final.
- e) Faça um algoritmo que calcule o volume de uma sala e escreva o resultado.
- f) Faça um algoritmo que calcule as quatro operações (+, -, * e /) entre dois números e escreva o resultado.
- g) Faça um algoritmo que solicite a área de uma parede (altura e largura), solicite a área de uma das faces de um tijolo e escreva quantos tijolos são necessários para construir a parede.

**APÊNDICE L - Lista de Exercícios: Comando Condicional SE (simples,
composto e encadeado)**

**RESOLVA OS EXERCÍCIOS A SEGUIR SOBRE O COMANDO CONDICIONAL SE
(SIMPLES, COMPOSTO E ENCADEADO)**

Exercite suas habilidades desenvolvendo os algoritmos solicitados a seguir:

- 1) Faça um algoritmo que leia uma letra e diga se esta equivale ao sexo feminino ou masculino.
- 2) Faça um algoritmo que leia dois números e escreva qual deles é o maior.
- 3) Faça um algoritmo que leia três números e escreva qual deles é o maior.
- 4) Faça um algoritmo que simule uma calculadora simples.
- 5) Faça um algoritmo que receba um número inteiro e verifique se este número é par ou ímpar.
- 6) Faça um algoritmo que leia um número e escreva se ele é: positivo, negativo ou zero.
- 7) Faça um algoritmo que leia o dia, o mês e o ano separadamente, apresente se esta data é válida ou não e escreva como no exemplo.
Exemplo: 25/08/2016.
- 8) Faça um algoritmo que leia dois número (X e Y), troque os valores entre as variáveis e escreva na tela os valores trocados.
- 9) Faça um algoritmo que leia um ano e determine se ele é bissexto.
- 10) Elabore um algoritmo que efetue o cálculo do reajuste de salário de um funcionário. Considere que o funcionário deverá receber um reajuste:
 - salário < 500 - 15%
 - salário >= 500 e <= 1000 - 10%
 - salário > 1000 - 5%
- 11) Faça um algoritmo que leia dois números. Se todos forem pares efetue a soma deles, caso todos sejam ímpares efetue a multiplicação e em caso contrário, subtraia um pelo outro. O algoritmo deve ainda emitir mensagens do tipo:
 - Todos pares, a soma é:
 - Todos ímpares, a multiplicação é:
 - Ambos os tipos, a subtração é:
- 12) Faça um algoritmo que leia a sigla de um estado da região sul e escreva o nome deste estado por extenso.
- 13) Leia um número real. Se o número for positivo imprima a raiz quadrada. Do contrário, imprima o número ao quadrado.

APÊNDICE M - Lista de Exercícios: Comando de Repetição PARA

RESOLVA OS EXERCÍCIOS A SEGUIR SOBRE O COMANDO DE REPETIÇÃO PARA

Exercite suas habilidades desenvolvendo os algoritmos solicitados a seguir, utilizando o comando de repetição PARA:

- 1) Faça um algoritmo que escreva os números de 1 a 10.
- 2) Faça um algoritmo que escreva os números de 1 a 1000.
- 3) Faça um algoritmo que escreva os números de 1 a 100, quebrando a linha de 10 em 10.
- 4) Faça um algoritmo que escreva os números pares entre 1 e 50. E, ao final, escreva a soma destes números.
- 5) Faça um algoritmo que leia 2 números inteiros e escreva os números que estão entre eles.
- 6) Analise o algoritmo a seguir e corrija o erro no código.

```

ALGORITMO "Exercicio06"
VAR
    numero1 : INTEIRO
    numero2 : INTEIRO
    soma1 : INTEIRO
    soma2 : INTEIRO
    contador : INTEIRO
INICIO
    PARA contador DE 1 ATE 4 FACA
        ESCREVA("Digite o 1º número: ")
        LEIA(numero1)
        soma1 <- soma1 + numero1
        ESCREVA("Digite o 2º número: ")
        LEIA(numero1)
        soma2 <- soma2 + numero2
    FIMPARA
    ESCREVA("A resposta é: ", soma1 / soma2)
FIMALGORITMO

```

- 7) Elabore um algoritmo que imprima a tabuada do 5.
- 8) Faça um algoritmo que leia 10 números inteiros, teste se são pares ou ímpares e escreva a soma dos pares e dos ímpares.

APÊNDICE N - Lista de Exercícios: Comando de Repetição ENQUANTO

RESOLVA OS EXERCÍCIOS A SEGUIR SOBRE O COMANDO DE REPETIÇÃO ENQUANTO

Exercite suas habilidades desenvolvendo os algoritmos solicitados a seguir, utilizando o comando de repetição ENQUANTO:

- 1) Faça um algoritmo que solicite números inteiros para o usuário, a cada número digitado pergunte se ele deseja continuar digitando.
- 2) Faça um algoritmo que escreva os números de 1 a 10, utilizando o ENQUANTO.
- 3) Faça um algoritmo que leia o número de nomes que o usuário terá que digitar.
Exemplo: - Quantos nomes deseja digitar? 2
 - Digite o nome 1:
 - Digite o nome 2:
- 4) Faça um algoritmo que leia vários números. O programa encerra quando for digitado o número zero. Escreva a soma dos números digitados.
- 5) Analise o algoritmo a seguir, escreva o que ele faz e caso apresente algum erro faça a correção.

```

ALGORITMO "Exercicio05"
VAR
    numero : INTEIRO
    contador : INTEIRO
    soma : INTEIRO
    resultado : INTEIRO
INICIO
    contador <- 0
    resultado <- 0

    ENQUANTO (numero >= 0) FACA
        ESCREVA("Informe um número: ")
        LEIA(numero)
        SE (numero >= 0) ENTAO
            contador <- contador + 1
            soma <- soma + numero
        FIMSE
    FIMENQUANTO

    resultado <- soma / contador
    ESCREVA("A média é: ", resultado)
FIMALGORITMO

```

- 6) Faça programa que simule uma calculadora simples, com as 4 operações.
- 7) Chico tem 1,50 m e cresce 2 centímetros por ano, enquanto Juca tem 1,10 m e cresce 3 centímetros por ano. Construir um algoritmo que calcule e escreva quantos anos serão necessários para que Juca seja maior que Chico.
- 8) Construa um algoritmo que leia uma quantidade indeterminada de números inteiros positivos e identifique qual foi o menor e maior número digitado. O algoritmo encerra quando for digitado um número negativo.

APÊNDICE O - O *Software VisuAlg* contribuiu para a aprendizagem de algoritmos

QUESTIONÁRIO - AVALIAÇÃO DA PRÁTICA PEDAGÓGICA

- 1) Qual forma de construção dos algoritmos você preferiu, digitando no editor de texto ou utilizando o *Software VisuAlg*? Quais as diferenças que você percebeu ao escrever os algoritmos das duas maneiras?
- 2) Aprender os algoritmos na Língua Portuguesa auxiliou ou não na compreensão dos códigos e comandos? Explique por que.
- 3) Como você julga o seu aprendizado da lógica de programação? Ao ler o enunciado dos problemas você consegue facilmente criar o algoritmo? Explique.
- 4) O uso do *Software VisuAlg* auxiliou ou não na compreensão e criação dos algoritmos? Explique por que.
- 5) Quais as dificuldades que encontrou no desenvolvimento das atividades e exercícios propostos. Explique.
- 6) Quais as dificuldades que você encontrou ao aprender em algoritmos durante as aulas? Explique detalhadamente.
- 7) Explique o que fazem os seguintes comandos:
 - a) LEIA -
 - b) SE -
 - c) PARA -
- 8) Explique o que faz o trecho do algoritmo a seguir:

```
teste <- "S"

ENQUANTO (teste = "S") FACA
  ESCREVA("Digite um número: ")
  LEIA(num)
  SE (num MOD 2 = 0) ENTAO
    teste <- "S"
  SENAO
    teste <- "N"
  FIMSE
FIMENQUANTO
```

- 9) Você acharia interessante que as aulas de algoritmos sempre utilizassem algum *software* em português para criar os códigos? Justifique sua resposta.
- 10) Avalie esta prática pedagógica descrevendo o que poderia ter sido melhor.



UNIVATES

Rua Avelino Tallini, 171 - Bairro Universitário
Lajeado | RS | Brasil | CEP 95914-014 | Cx. Postal 155
Telefone: (51) 3714-7000
www.univates.br | 0800-700-809